Korbinian Schreiber, Heidelberg University

accelerated neuromorphic cybernetics

Inaugural-Dissertation

zur Erlangung der

Doktorwürde

der

Naturwissenschaftlich-Mathematischen Gesamtfakultät

der Ruprecht-Karls-Universität

Heidelberg

Vorgelegt von

Korbinian Schreiber

geboren in München, Deutschland

Tag der mündlichen Prüfung: 9. Dezember 2020

The cover image shows 17 outbound and return trajectories of an accelerated neuromorphic insect. A detailed description is part of section 5.6.

Beschleunigte neuromorphe Kybernetik

Referees:
Dr. Johannes Schemmel (Universität Heidelberg)
Prof. Dr. Michael Hausmann (Universität Heidelberg)

Dissertation

submitted to the
Combined Faculty of Natural Sciences and Mathematics
of Heidelberg University, Germany
for the degree of
Doctor of Natural Sciences

Put forward by
Korbinian Schreiber
born in Munich, Germany

Oral examination: 2020 December 9th

Accelerated neuromorphic cybernetics

Referees:

Dr. Johannes Schemmel (Heidelberg University)

Prof. Dr. Michael Hausmann (Heidelberg University)

Beschleunigte neuromorphe Kybernetik:

Beschleunigte analog-digitale neuromorphe Hardware beschreibt elektronische Systeme, die elektrophysiologische Aspekte biologischer Nervensysteme auf beschleunigte Weise in analogen Spannungen und Strömen emulieren. Während die funktionale Bandbreite dieser Systeme bereits viele beobachtete neuronale Fähigkeiten, wie etwa Lernen oder Klassifizieren, abdeckt, bleiben einige Bereiche noch weitgehend unerschlossen. Mitunter betrifft dies kybernetische Szenarien, in denen Nervensysteme in geschlossener Interaktion mit ihren Körpern und Umgebungen stehen. Da die Steuerung von Verhalten und Bewegungsabläufen in Tieren sowohl den Zweck als auch die Entstehungsursache von Nervensystemen darstellt, sind solche Prozesse in der Natur jedoch von essentieller Bedeutung. Neben dem Entwurf neuromorpher Schaltkreis- und Systemkomponenten ist der Hauptgegenstand dieser Arbeit deshalb die Konstruktion und Analyse beschleunigter neuromorpher Wesen, die in kybernetische Wirkungsketten eingebunden sind. Dabei handelt es sich zum einen um die Beschreibung eines beschleunigten mechanischen Roboters, zum anderen um ein beschleunigtes virtuelles Insekt. In beiden Fällen sind die sensorischen Organe, sowie die Aktuatorik ihrer künstlichen Körper aus der Neurophysiologie der biologischen Vorbilder abgeleitet und so originalgetreu wie möglich nachgebildet. Darüber hinaus werden die beiden biomimetischen Organismen jeweils einer evolutionären Optimierung unterzogen, die die Vorteile beschleunigter neuromorpher Nervensysteme durch signifikante Zeitersparnis verdeutlicht.

Accelerated neuromorphic cybernetics:

Accelerated mixed-signal neuromorphic hardware refers to electronic systems that emulate electrophysiological aspects of biological nervous systems in analog voltages and currents in an accelerated manner. While the functional spectrum of these systems already includes many observed neuronal capabilities, such as learning or classification, some areas remain largely unexplored. In particular, this concerns cybernetic scenarios in which nervous systems engage in closed interaction with their bodies and environments. Since the control of behavior and movement in animals is both the purpose and the cause of the development of nervous systems, such processes are, however, of essential importance in nature. Besides the design of neuromorphic circuit- and system components, the main focus of this work is therefore the construction and analysis of accelerated neuromorphic agents that are integrated into cybernetic chains of action. These agents are, on the one hand, an accelerated mechanical robot, on the other hand, an accelerated virtual insect. In both cases, the sensory organs and actuators of their artificial bodies are derived from the neurophysiology of the biological prototypes and are reproduced as faithfully as possible. In addition, each of the two biomimetic organisms is subjected to evolutionary optimization, which illustrates the advantages of accelerated neuromorphic nervous systems through significant time savings.

Contents

1 Motivation **1**
1.1 Agents and cybernetics 1
1.2 Insect brains 5

2 Background **11**
2.1 Biology 11
2.1.1 Shape of nervous systems 11
2.1.2 Dynamics of neurons 14
2.1.3 Dynamics of synapses 23
2.1.4 Biological cybernetics 26
2.2 Neuromorphic systems 29
2.2.1 von Neumann architectures 29
2.2.2 Computational complexity of brains 30
2.2.3 The von-Neumann brain 31
2.2.4 Parallel computation and memory 32
2.2.5 Neuromorphic devices 32
2.2.6 BrainScaleS 35

3 BrainScaleS-2 **37**
3.1 The prototype chips 38
3.1.1 Synapse drivers and synapses 39
3.1.2 Neurons 40
3.1.3 Correlation ADC 42
3.1.4 Analog parameter storage 42
3.1.5 Membrane ADC 43
3.1.6 Analog readout 43
3.1.7 Digital neuron back end 43
3.1.8 Phase-locked loop for internal clock generation 44
3.1.9 Communication and FPGA board 44
3.1.10 Summary 44
3.2 Prototype test setup 45
3.2.1 Power supply 45
3.2.2 MADC test input 46
3.2.3 ADC chain 47
3.2.4 Reference signals 47
3.2.5 Spike I/O 47
3.2.6 Spike router 47
3.3 HICANN-X 49
3.3.1 ANNCORE and quadrants 49

3.3.2 Other components . . . 51
3.4 Full-scale test setup . . . 52
3.4.1 Power supply . . . 52
3.4.2 Readout chain and reference generation . . . 54
3.4.3 Digital I/O . . . 54
3.4.4 Chip carrier . . . 55
3.5 CADC . . . 57
3.5.1 CADC center . . . 57
3.5.2 Columns and channels . . . 58
3.5.3 Measurements and performance . . . 58
3.5.4 Discussion and outlook . . . 65

4 Accelerated mechanics 67
4.1 Motivation . . . 67
4.2 Requirements and limitations . . . 68
4.2.1 Actuators . . . 69
4.2.2 Sensors . . . 72
4.3 PlayPen . . . 77
4.3.1 Actuators . . . 77
4.3.2 Sensors . . . 81
4.3.3 Prototype . . . 85
4.4 Edge tracing . . . 91
4.5 Maximum finder on prototype . . . 93
4.6 PlayPen-2 . . . 101
4.6.1 Hardware improvements . . . 102
4.6.2 Sensor head . . . 104
4.6.3 Rotary sensors . . . 105
4.6.4 Microcontroller firmware . . . 106
4.7 Maximum finder on second revision . . . 111
4.8 Evolutionary optimization . . . 114
4.9 Outlook . . . 116

5 Accelerated insects 119
5.1 The central complex . . . 120
5.2 Path integration . . . 122
5.3 A neural model for path integration . . . 123
5.3.1 Subpopulations and network topology . . . 124
5.3.2 Abstract connectivity . . . 127
5.3.3 Sensors . . . 129
5.3.4 Integrators . . . 131
5.3.5 Motoric outputs . . . 132
5.4 LIF-based model . . . 132
5.4.1 From spike rates to spikes . . . 132
5.4.2 General neuron properties . . . 133

5.5 Insects on BrainScaleS-2 134
5.5.1 Non-integrating neurons 134
5.5.2 Integrator neurons 137
5.5.3 CPU4 weight access 139
5.5.4 Reduced network size 139
5.5.5 Sensor inputs 140
5.5.6 Motor output 140
5.5.7 Experiment execution and scheduling 140
5.6 Execution and parameter sweeps 142
5.6.1 Parameter sweep 143
5.7 Evolutionary optimization 147
5.8 Discussion and outlook 148

6 Concluding remarks 151

7 Bibliography 157

Glossary 179
7.1 List of Figures 181
7.2 List of Tables 184

I Appendix 185

A Additional figures 187
A.1 Insect trajectories 187

B Code snippets 193
B.1 CPU4 read access 193
B.2 CPU4 write access 195

C Schematics 197
C.1 Baseboard for the 3rd generation prototype 198
C.2 xBoard 202
C.3 CADC 207
C.3.1 Channel 207
C.3.2 Comparator 207
C.3.3 Channel (digital) 208
C.3.4 Ramp generator 208
C.4 PlayPen-1: core module 209
C.5 PlayPen-1: signal processing module 211
C.6 PlayPen-1: sensor head 213
C.7 PlayPen-2 214
C.8 PlayPen-2: rotary sensor 216
C.9 PlayPen-2: sensor head 217

1 Motivation

"Oddly enough, it was only in 2040 that the informationists, cipher theorists, and other experts expressed surprise at how their predecessors could have been so blind for so long, struggling to create artificial intelligence. After all, for the overwhelming majority of tasks performed by people in 97.8 percent of both blue- and white-collar jobs, intelligence was not necessary. What was necessary? A command of the situation, skill, care, and enterprise."

– Stanisław Lem, Weapon systems of the 21st century or The upside-down evolution, 1983

1.1 Agents and cybernetics

In his book, Stanisław Lem writes from a fictitious future perspective about the developments of artificial intelligence in the 21$^{\text{th}}$ century. What can be attested 37 years after the story was published is that scientists and engineers are still struggling to create true artificial intelligence and that a vast number of jobs indeed require "[a] command of the situation, skill, care, and enterprise". Debatable, however, is perhaps, whether intelligence may not be precisely that essential prerequisite for the listed aspects that Lem contrasts it with. The etymology of the term intelligence reveals a Latin origin from the word *intelligentia* that describes insight and understanding, and thus, a rather abstract class of anthropogenic mental phenomena. Given Lem's intellectual background (Lem, 1981), this is probably where the distinction between the aforementioned aspects of practical action and intelligence arises from. Seen from another perspective though, the distinction may fade:

Having insight or understanding of a phenomenon is usually claimed when a set of representations (either formal or mental) have been found that are precise enough to reliably reconstruct, i.e. predict, the phenomenon. Information on the phenomenon must therefore somehow be communicated to the understander, and the understander must somehow be able to communicate his understanding back in order to prove it. In the context of the natural sciences, the evidence for understanding must be given formally, i.e., symbolically. A football player on the other hand can prove his understanding of the game by his skills on the playing field. Insofar, understanding or insight is always related to communication, i.e., either physical action, formal language, or at the very least thinking.

Language, however, is formed by information exchange between individuals, which, in turn, relies on physical processes.

Logical thinking can be regarded as an interaction of the mind (the thinking entity) with abstract ideas and their consequences. In a way, vocabulary and grammar are to the mind, what objects and forces are to the hands.

The spirited term *intelligence* is therefore intrinsically related to interaction, either formally, mentally, or physically, while the distinction between the three is not so relevant from a high-level perspective. Norbert Wiener referred to the science and phenomena related to "control and communication in the animal and the machine" as *cybernetics* (Wiener, 1948) instead. The neo-Greek expression derives from *kybernetes* (steersman) and means steersmanship. To maneuver a vessel, a steersman must be versed in physically controlling the rudder, but also in mastering symbolic navigation. Unlike intelligence, cybernetics directly implies an exchange of information beyond mere understanding. One could say that cybernetics describes the application of intelligence, in terms of recognition and action, as well as intelligence itself. In Lem's story, it would comprise both, symbolic artificial intelligence, as well as what is necessary for the other 97.8 percent.

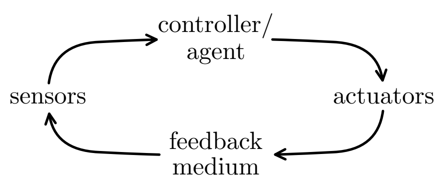


Figure 1.1: A cybernetic loop.

According to Wiener (1948), a cybernetic situation always implies a control loop that is formed by an acting *agent* (steersman), a set of actuators (rudder), a feedback medium (water and vessel), and a set of sensors (eyes, compass). In neuroscience, this exact scheme became known today as a *closed-loop* situation (Ejaz et al., 2013; Potter et al., 2014; El Hady, 2016; Zhao et al., 2020). Why the much more precise, and already well-defined term *cybernetics* became forgotten and replaced by a more ambiguous, indeclinable, and awkward word compound appears somewhat nebulous. Cariani (2010) reports that the two movements artificial intelligence and cybernetics "coexisted for roughly a decade, but by the mid-1960s, the proponents of symbolic AI gained control of national funding conduits and ruthlessly defunded cybernetics research. This effectively liquidated the subfields of self-organizing systems, neural networks and adaptive machines, evolutionary programming, biological computation, and bionics for several decades, leaving the workers in management, therapy and the social sciences to carry the torch." Be that as it may, in this thesis, the term *cybernetics* is extensively employed for its semantic excellence and grammatical superiority. In suitable situations, however, *closed-loop* is used as well.

Cybernetic experiments that involve *neuromorphic* agents are at the center of this thesis. *Neuromorphic* is another neo-Greek expression that roughly translates to "shaped like nerves". The term was introduced by Carver Mead to describe large scale integrated electronic circuits that mimic networks of biological neurons (Mead, 1990). The neuromorphic system, on which the experiments presented in this thesis were carried out, is the second generation of the BrainScaleS system. BrainScaleS-2 is a multi-scale neuromorphic computing substrate that is being developed at

Heidelberg University within the Human Brain Project (HBP). While it is covered in extensive detail in later section 2.2 and chapter 3, it is relevant to note here that BrainScaleS-2 is substantially a family of mixed-signal microchips. I.e., both analog and digital electronic integrated circuits are combined to mimic biological neurons and synapses in a way that faithfully reproduces their dynamic behavior. Especially, the membrane potential, i.e. the voltage across a neuron's cell membrane, and the spike-based communication are reproduced (see section 2.1).

In addition, the chip's analog neural networks are tightly coupled to a conventional digital processing unit, on which arbitrary control or learning algorithms can be implemented and executed in parallel to the artificial neural processes. Moreover, the network dynamics evolve significantly faster as compared to biological cells, that is to say, in an *accelerated* way.

Besides research on *accelerated neuromorphic cybernetics*, as the document's title promises, parts of this thesis also involve engineering on BrainScaleS-2. In particular, a highly parallel analog-to-digital converter (ADC) that is involved in on-chip learning and inference tasks has been designed and implemented. Based on the data collected by the ADC learning or calibration algorithms can take decisions and actions on the network parameters. The ADC is therefore designed to take part in another class of cybernetic control loops: In this context, they are formed between the on-chip digital processor (agent) that takes action on the network parameters (actuators) which, in turn, influence the neural network (feedback medium) what is finally sensed by the ADC (sensor). The technical details together with some evaluation and experimental results are part of section 3.5.

Unlike tightly specified conventional engineering projects, neuromorphic systems are being developed together with the understanding of the substrate that they emulate. The more capable neuromorphic systems become, the more can be learned about both, their own function, as well as the emulated biological substrate. For example, during the last five years, i.e., the period of the author's doctoral studies, multiple paradigm shifts concerning learning rules for spiking neural networks occurred. Most importantly, the successes of what is commonly known as *deep learning* (Schmidhuber, 2015) spilled over from the application-oriented field of machine learning to the rather academic discipline of computational neuroscience. In particular, the core-concept of deep learning, i.e. error backpropagation, has since been translated in many different ways from continuous artificial to spiking neural networks (Lee et al., 2016; Sacramento et al., 2018; Zenke and Ganguli, 2018; Bellec et al., 2019, 2020).

However, most of these theoretical advances are not specifically concerned with cybernetic agent-environment systems. I.e., situations where a nervous compound takes some sort of control over an external system from which it receives feedback according to its actions. Instead, the majority of them assumes feedforward situations, where data enters the neural network and leaves it in a processed state, which is then evaluated by the experimenter or the learning rule. The data might be pictures, videos, or text, and the output might be labels (in classifying networks), or again pictures, videos, or text (in autoencoders, or transformer networks). In these cases, from a cybernetic perspective, the network can be regarded as the feedback medium, its quantified performance as the sensor signal, and the learning rule as the agent who is taking action on the network parameters. While this architecture is optimized for studying learning rules, it tends to bring neural networks into unnatural situations.

First, neural networks (in nature) always operate in feedback scenarios. Even a single nerve cell is always part of a feedback loop, as its action causes effects in the surrounding network or muscle tissue that necessarily reflect back at it in some way (Brette, 2013; El Hady, 2016). This behavior is partly covered in recurrent neural networks but lacking in typical multilayer feedforward networks. Second, natural neural networks do not operate in a batch-like manner. I.e., they are not consecutively switched on for the time of the classification or generation and then switched off again for parameter updates, but they can operate continuously and are observed to do both, inference and learning, in parallel.

However, there are of course also research areas that are concerned with more organic and biologically plausible embeddings of neuronal networks. One example is so-called online learning rules that apply parameter updates while the networks are actively classifying. Although these rules contribute to the problem of parallel inference and learning, they do not specifically address the agent-environment loop.

Reinforcement learning would be a counterexample where this loop is of central interest (Frémaux et al., 2010, 2013; Frémaux and Gerstner, 2016; Wunderlich et al., 2019). Here, interacting agents are observed for a certain amount of time and afterward evaluated for their performance. A scalar reward signal is then incorporated into the learning rule that usually updates the network parameters after several evaluation cycles.

Apart from learning scenarios, agent-environment loops are also at the focus of neurorobotics. Here, the goal is to employ neural substrates for controlling physical or virtual robots, with or without learning. General examples can be found in Doya and Uchibe (2005); Ijspeert et al. (2007); Rucci et al. (2007); Cox and Krichmar (2009); Manoonpong et al. (2013); Vannucci et al. (2015); Stone et al. (2017); Knight et al. (2019). The resulting kinetics of these robots, however, are in almost all cases on the same time scale as the biological paragons. I.e., their movements resemble those of biological agents in terms of speed and acceleration.

In contrast, the neurorobotic experiments, which are part of chapter 4, cope with the thousandfold acceleration of BrainScaleS-2. For that purpose, a specialized but still versatile robotic platform has been developed that contains interfaces for converting neural signals into motor commands and sensor information into neural signals. The design and construction process of this setup, as well as several experiments, are outlined in this thesis.

Apart from this physical robotic agent, a virtual cybernetic creature is introduced in chapter 5. In this experiment, a bee-like agent has to return to its nest after foraging in a simulated environment. The insect's body, actuators, and sensors are implemented in accelerated real-time on the digital coprocessor of BrainScaleS-2, while an anatomically-derived network model runs in parallel on the chip's analog core. Despite being extremely small (even for insect standards), this neuromorphic brain masters its task robustly and with precision.

According to Lem, not only navigation, but all the other qualities that are necessary for the "blue- and white-collar jobs", i.e., "[a] command of the situation, skill, care, and enterprise" can "be found in insects" too.

1.2 Insect brains

"The wasp may have enough nerve tissue to drive a truck from a port to a distant city or to guide a transcontinental rocket. It is only that its nervous system was programmed by natural evolution for completely different tasks. Successive generations of information theorists and computer scientists had labored in vain to imitate the functions of the human brain in computers; stubbornly they ignored a mechanism a million times simpler than the brain, incredibly small, and remarkably reliable in its operation. Not artificial intelligence but artificial instinct should have been simulated for programming at the outset."
– Stanisław Lem, Weapon systems of the 21st century or The upside-down evolution, 1983

What Lem refers to by *artificial instinct* is, in terms of the above considerations and opposed to symbolic intelligence, the intelligence behind action. Again, it can be attested 37 years after Lem's text appeared, that "successive generations of information theorists and computer scientists had labored in vain to imitate the functions of the human brain in computers". And again, an aspect of his prognosis remains debatable: whether or not insect brains have been ignored in research since then.

To get hints about certain trends or interests in popular as well as in scientific literature, the Google Books Ngram Viewer (Michel et al., 2011) can be consulted. This online service extracts the relative occurrence of certain terms or word combinations in the published literature from Google's vast archive of millions of digitized books. The data for 'human brain' and 'insect brain' is plotted in figure 1.2. After the interest in both topics increased from the end of the 1940s until around 1965, only the interest in 'human brain' continued to grow at about the same pace. The appearance of 'insect brain' stagnated instead. Lem's prediction might therefore hold to a certain extent, at least as far as the published interest in the two topics in relation to each other is concerned.

Further evidence is that the number of neurons in brain simulations, in particular those that concern mammalian brains, continuously increased over time reflecting a certain amount of effort going into that direction. There are manifold motivations behind this, of which four important ones are listed below.

First, there are brain areas that intrinsically require high dimensional processing capabilities, like for example the visual cortex (see section 4.2.2). Without large network simulations, these areas are hard to study in a biologically plausible manner.

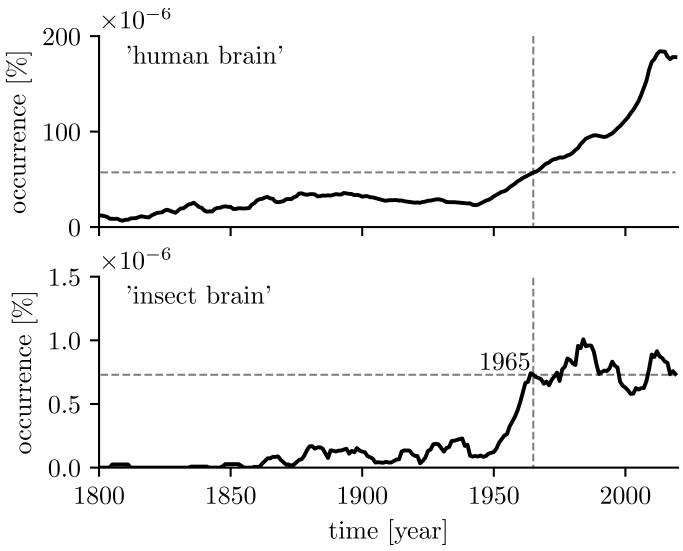


Figure 1.2: The relative occurrence of the word combination 'human brain' and 'insect brain' in literature. While 'insect brain' stagnated from 1965 on, 'human brain' approximately tripled in its relative occurrence. Data gathered from Google Books Ngram Viewer (Michel et al., 2011).

Second, a cortical area of only $1\,\mathrm{mm}^2$ contains already $\sim \mathcal{O}(10^5)$ neurons and $\sim \mathcal{O}(10^9)$ synapses (Keller et al., 2018). Due to this high neuron density, even small mammalian brains, like those of rodents, consist of $\sim \mathcal{O}(10^8)$ neurons (Herculano-Houzel et al., 2006), which is still a very challenging number for today's simulation engines.

Third, pushing for large-scale brain simulators is a practical way to incentivize the development of novel computing technologies in a goal-driven way (section 2.2). E.g., the HBP as one of the Future and Emerging Technology Flagships of the European Union relies partly on this strategy. Also companies like Intel and IBM recently invested large-scale efforts into that direction (Merolla et al., 2014; Davies et al., 2018).

Fourth, practical applications in machine learning can evidently benefit from larger networks. Particularly in the recent decade, many nontrivial problems were solved predominantly because a larger amount of computing resources became available (He et al., 2015; Radford et al., 2019; Brown et al., 2020). Among those successes were some that were predicted to occur at least a decade later (Bostrom, 2017; Silver et al., 2016, 2017; Vinyals et al., 2019).

However, achievements, which have to the largest amount resulted from a sheer increase in available computing resources (winning Go, beating Humans in image classifications, etc.), currently likely outweigh the achievements that should be aimed for in a scientific context: insight and understanding. Some researchers in artificial intelligence are even expecting general artificial intelligence (which mostly refers to one that passes the Turing test (Turing, 1950)) to miraculously emerge from upscaled networks (Bostrom, 2017; Koch, 2019), implying that the problem has already been solved and is only waiting for its substrate.

So far, however, machine learning revealed a huge potential in terms of symbolic artificial intelligence but also a tremendous lack of practical intelligence. I.e., most of these networks achieve superhuman performance in pattern recognition but they continuously produce errors that reflect their misunderstanding of the actual underlying structure (Nguyen et al., 2015; Moosavi-Dezfooli et al., 2016; Papernot et al., 2016; Geirhos et al., 2017). For example Su et al. (2019) showed that a number of image classifiers could reliably be fooled by single-pixel attacks. E.g., they recognized a frog instead of a horse, because one pixel was manipulated.

Other examples were generated by GPT-3 (Brown et al., 2020), a generative language model that is based on a gigantic neural network with 175 billion parameters. The following excerpt from a tale written by GPT-3 about some mice living at the Yangtze river (gwern.net and GPT-3, 2020) might illustrate the type of confusion these networks have between the mere structure of language, which they master at superhuman performance, and the underlying reality, that they do not seem to protrude: "[...] While they were swimming, another giant fish appeared. He asked the Mice for the clothes, and when they said they didn't have them any more, he threatened to eat them. So, the Mice swam away as fast as they could. And while they were swimming, another giant fish appeared. This time it was a tiger. [...]"[1] So while the network can spin an exciting story, even using rhetorical elements, it appears to not realize that fish do not suddenly emerge as tigers usually.

Recently, Hermann Haken, the founder of *synergetics* (Haken, 1980), a theory about self-organization, both in dead and living matter (Haken and Haken-Krell, 1989), was asked why synergetics is yet ignored by the majority of the machine learning and computational neuroscience community (Haken, 2018). Haken answered, "What people are doing, I have to say, is that they are always building a certain model, [...] but there are no penetrating concepts that you can see. [...] So I have to say that, in my opinion, there is too much tinkering and too little theory. Or take Deep Learning today, with all these layers. It's like training a wild animal: in many cases, it works, but sometimes it bites. And especially with Deep Learning, nobody knows how it works."[2] The sudden tiger in GPT-3's story, might be one of the wild animals that Haken was referring to.

Let us turn back the focus on biological networks. Similar to machine learning, there is a trend to scale up network sizes to search for emergent phenomena or to reconstruct what is being observed in biology (Kunkel et al., 2014; Jordan et al., 2018; Igarashi et al., 2019). While research on these large-scale networks is of course justified and evidently produces outstanding results, for the most part, it confirms what is already known or suspected. Its main yield, so far, is to confirm existing theories (Reimann et al., 2013; Markram et al., 2015).

[1]The author of the current document warmly recommends enjoying a few of the stories of GPT-3 on the above-referenced website.

[2]This quote has been translated by the author of this thesis from what Haken said in German. The actual quote is "Was die Leute da machen, muss ich leider sagen, ist, dass sie immer nur ein bestimmtes Modell basteln, [...], aber es gibt keine durschschlagenden Konzepte, die man sieht. [...] Ich muss also schon sagen, dass in meinen Augen da zu viel Bastelei dabei ist und zu wenig Theorie. Oder nehmen Sie Deep Learning heutzutage, mit diesen vielen Schichten. Das ist so als würde ich ein wildes Tier dressieren: in vielen Fällen funktioniert es, aber manchmal beißt es doch. Und gerade beim Deep Learning weiß kein Mensch, wie's funktioniert." The question has been posed by Yulia Sandamirskaya, who is also conducting research on cybernetic neuromorphic agents (Sandamirskaya et al., 2011; Milde et al., 2017) at the University and ETH Zurich.

On the other hand, it appears that computational neuroscience that focuses on functional principles in smaller-scale networks ($\sim \mathcal{O}(10^3)$ neurons), continuously achieves valuable scientific output in terms of new theories. Especially research on learning rules (Neftci et al., 2014; Zenke and Ganguli, 2018; Sacramento et al., 2018; Bellec et al., 2019), and architectural and functional models (Stone et al., 2017; Masoli et al., 2020; Müller et al., 2020; Zhao et al., 2019) relies mostly on small-scale simulations. But while there is in fact a lot of research on small networks, there still lies an ocean of unresearched phenomena calm and often only softly touched by funding at scales that are already long within the reach of classical and neuromorphic computing (Turner-Evans and Jayaraman, 2016; Sarma et al., 2018; Xu et al., 2020). Moreover, it must be recognized that networks with only $\mathcal{O}(100)$ neurons remain far from being well-understood (Leng et al., 2018; Cramer et al., 2020b; Haluszczynski et al., 2020). As long as that is the case, does it not seem much more obvious to start with small, relatively simple brains before tackling the most complicated ones?

Lem's prediction about misguided research on the human brain, therefore, does not seem completely audacious, but it is certainly not entirely accurate either. Considering *artificial instinct* though, one could still argue that at least a certain imbalance in favor of feedforward instead of feedback loops still exists today. However, for some years now also this gap is closing gradually, mainly driven by biological closed-loop experiments with small mammals, fish, or insects. An impressive example was given recently (Huang et al., 2020) by experiments on larval zebrafish which are so transparent that their brains can be imaged noninvasively. Furthermore, the fish can genetically be modified to express fluorescent proteins that are sensitive to calcium concentrations. Since neural membrane potentials are intrinsically bound to calcium concentrations, this fluorescence directly reflects neural activity. Using fluorescence microscopy, this method can allow for recording the entire brain activity of a fish at single-cell level accuracy in real-time (Ahrens et al., 2013).

Huang et al. (2020) used fast digital image analysis to infer the intended motion of a fish that was tethered for this type of imaging. They used these movement signals to steer a simulated agent through a virtual environment. Similar to a video game, the simulation produced visual feedback that was presented to the tethered animal via computer screens, effectively creating a virtual reality for zebrafish. In this way, both, the animal's behavior as well as its cell-level brain-wide activity can be observed and interacted with in real-time.

This work aims at developing a deeper understanding of small-scale networks on analog neuromorphic hardware. In addition, it seeks to explore the immense capabilities of such miniature brains with a specific focus on biologically inspired cybernetic agents. On the one hand, this is pursued with the above mentioned robotic platform that combines BrainScaleS-2 with emulated nervous sensors and motor organs into an accelerated electromechanical organism.

On the other hand, this approach is taken in chapter 5 by implementing a fully autonomous insectoid agent that uses a neuromorphic nervous system to spatially navigate through a virtual environment. This experiment confirms that, when small-scale neural arrangements are considered carefully, only 17 neurons can be sufficient to make an artificial bee find its way back to its nest.

This PhD thesis is intended as a contribution to the transition from widespread symbolic artificial intelligence to skilled artificial instinct before the year 2040.

2 Background

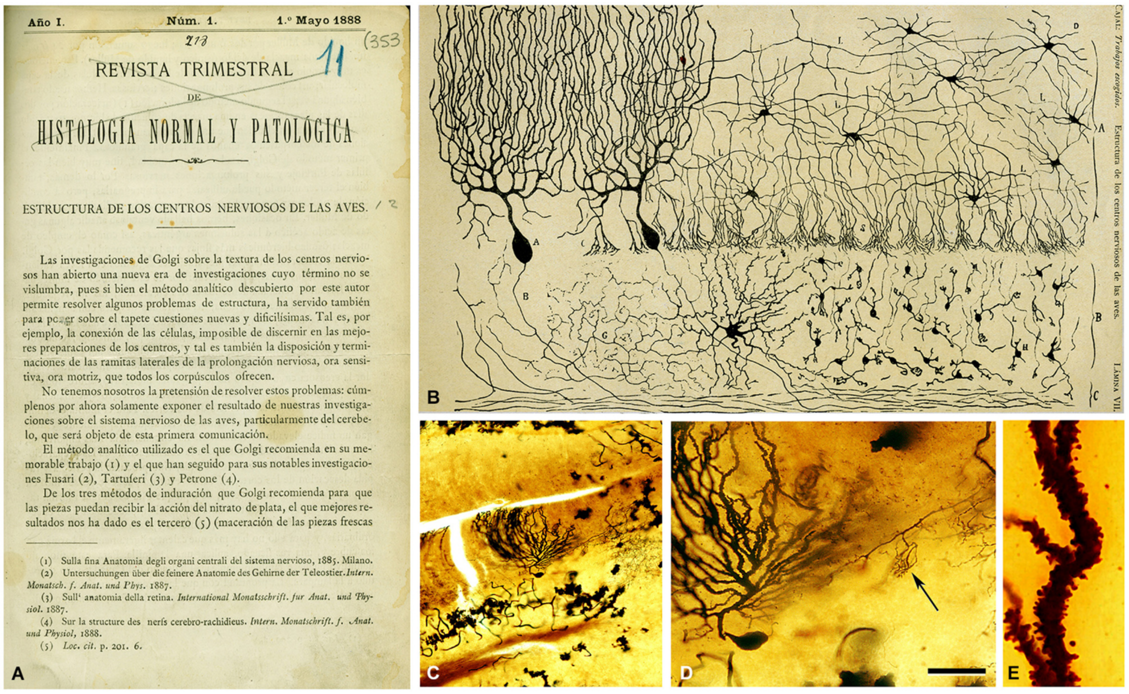
"Intelligent life on a planet comes of age when it first works out the reason for its own existence."
– Richard Dawkins

2.1 Biology

2.1.1 Shape of nervous systems

The first considerably detailed pictures of nervous systems became available with an improved sliver staining technique presented by Camillo Golgi in 1873 (Finger, 2001; Grant, 2007). Golgi himself first interpreted what he saw not as a separated but as a continually connected giant structure of delicately branched cell tissue. A few year later, Ramon y Cajal discovered that this immense web was indeed separated into many single cells (y Cajal, 1888; Cimino, 1999). y Cajal (1888) provided visual evidence, collected by means of Golgi's staining method, that the nerve cells were not continuous but separated in the brains of adult birds (DeFelipe, 2015). Figure 2.1 shows the title page of this seminal article, some of the micrographs that were obtained in his measurements, and a few of Cajal's famous drawings. The term *neuron* was coined a little later by Waldeyer (1891) (Scheuerlein et al., 2017).

Figure 2.2 gives a more detailed and contemporary schematic illustration of a typical neuron and how it is connected to its surrounding. Like most other animal cells, also neurons contain the basic eucaryotic set of organelles and proteins that are relevant for its metabolism, structure and life cycle (Alberts et al., 2007). Unlike other cells, however, neurons have a very unique shape and some special features related in particular to their cell membrane. Around their cell body, called the *soma*, unfold many long and finely branched structures, the *dendrites*. Together, these dendrites constitute the *dendritic tree*, which provides the primary membrane area for other neurons to dock at. On one side of the neuron, at the *axon hillock*, the cell membrane tapers into a thin tube, the *axon*, which can extend over distances from about 0.1 mm to 2 m in the case of humans (Kandel et al., 2013). The axon can also branch out but usually not as much as the dendrites. The primary function of the axon is to transport information in terms of time-varying electrical potentials from the soma to other neurons at potentially distant locations. It is surrounded by myelin sheaths, which grow out of the so-called Schwann cells. These sheaths provide increased electrical insulation between the cytosol and the extracellular space, resulting in a substantially increased transmission rate for the electrical signals (Kandel et al., 2013). The axon docks at other neurons via *synapses*.



Año I. Núm. 1. 1.° Mayo 1888

218 (353

11

~~REVISTA TRIMESTRAL~~

~~DE~~

~~HISTOLOGÍA NORMAL Y PATOLÓGICA~~

ESTRUCTURA DE LOS CENTROS NERVIOSOS DE LAS AVES. / 2

Las investigaciones de Golgi sobre la textura de los centros nerviosos han abierto una nueva era de investigaciones cuyo término no se vislumbra, pues si bien el método analítico descubierto por este autor permite resolver algunos problemas de estructura, ha servido también para poner sobre el tapete cuestiones nuevas y dificilísimas. Tal es, por ejemplo, la conexión de las células, imposible de discernir en las mejores preparaciones de los centros, y tal es también la disposición y terminaciones de las ramitas laterales de la prolongación nerviosa, ora sensitiva, ora motriz, que todos los corpúsculos ofrecen.

No tenemos nosotros la pretensión de resolver estos problemas: cúmplenos por ahora solamente exponer el resultado de nuestras investigaciones sobre el sistema nervioso de las aves, particularmente del cerebelo, que será objeto de esta primera comunicación.

El método analítico utilizado es el que Golgi recomienda en su memorable trabajo (1) y el que han seguido para sus notables investigaciones Fusari (2), Tartuferi (3) y Petrone (4).

De los tres métodos de induración que Golgi recomienda para que las piezas puedan recibir la acción del nitrato de plata, el que mejores resultados nos ha dado es el tercero (5) (maceración de las piezas frescas

(1) Sulla fina Anatomia degli organi centrali del sistema nervioso, 1885. Milano.

(2) Untersuchungen über die feinere Anatomie des Gehirne der Teleostier. *Intern. Monatsch. f. Anat. und Phys.* 1887.

(3) Sull' anatomia della retina. *International Monatsschrift. fur Anat. und Physiol.* 1887.

(4) Sur la structure des nerfs cerebro-rachidieus. *Intern. Monatschrift. f. Anat. und Physiol*, 1888.

(5) *Loc. cit.* p. 201. 6.

A
B
C
D
E
CAJAL: Trabajos escogidos. Estructura de los centros nerviosos de las aves.
LÁMINA VII.

Figure 2.1: First illustration by Cajal (1888) of a Golgi impregnated preparation of the nervous system. (A) First page of the article and (B) illustration whose legend states: "Vertical section of a cerebellar convolution of a hen. Impregnation by the Golgi method. A represents the molecular zone, B designates the granular layer and C the white matter" . (C) photomicrograph from one of Cajal's preparations of the cerebellum of an adult bird stained with the Golgi method. (D) higher magnification of (C) to illustrate a Purkinje cell and a basket formation (arrow). (E) dendrite of the Purkinje cell which is covered with dendritic spines. The histological images were obtained by Pablo García-López, Virginia García-Marín, and Miguel Freire (Legado Cajal, Instituto Cajal). Scale bar: 200 µm in (C); 60 µm in (D); 8.4 µm in (E). Based on y Cajal (1888) and Oroquieta (2014). Figure and description are taken from DeFelipe (2015). (CC BY).

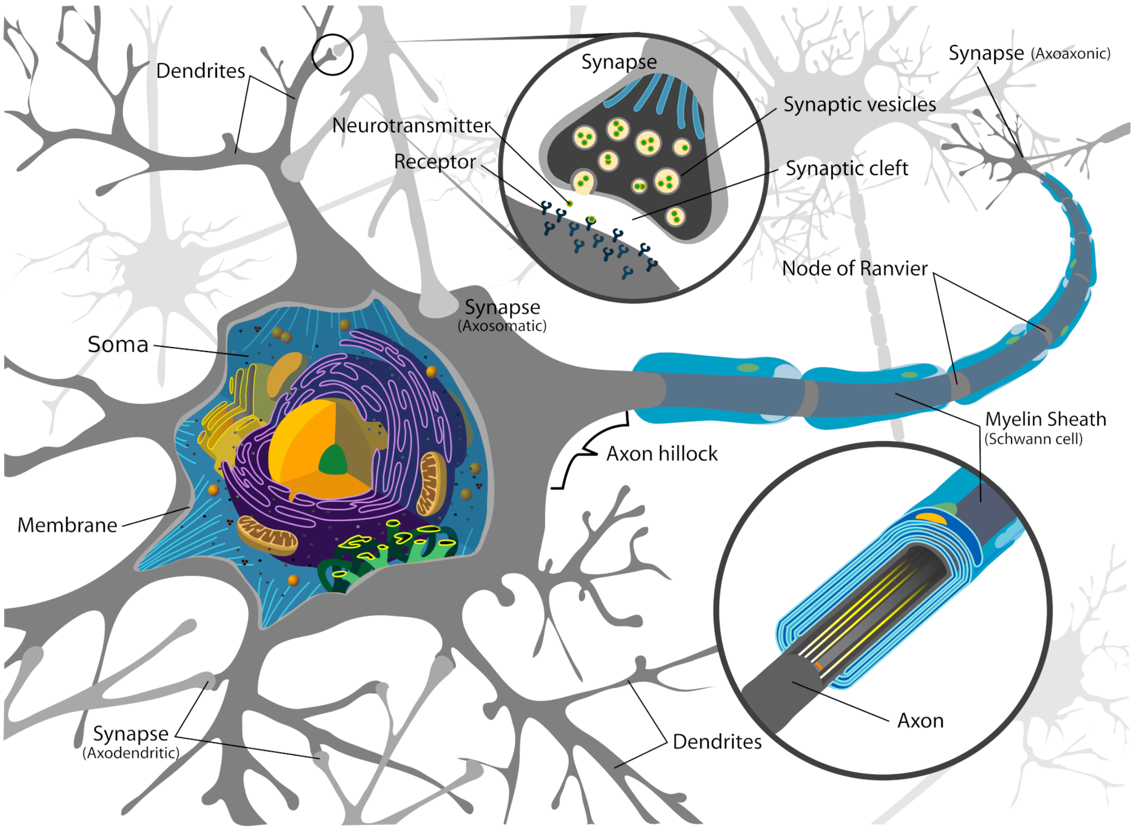


Figure 2.2: Detailed illustration of a neuron surrounded by other neurons. The most important components for neural information processing are labeled. Based on LadyofHats (2007).

The synapse comprises the cellular and extracellular space at the location where the two neurons meet. It contains the *synaptic cleft*, which is the small volume in between, and the neighboring cell volumes of both, the pre- and postsynaptic neuron.

All models and descriptions within this thesis implicitly assume the *neuron doctrine*. I.e., it is assumed that the relevant part of nervous information processing is carried out by neurons. Moreover, it is assumed that the relevant messaging between different neurons happens only synaptically and via spike based transmission, if not otherwise noted. The next section describes phenomena that are in relation to the dynamics of the membrane potential.

2.1.2 Dynamics of neurons

Research on the electrical properties of biological tissue dates back to the late 18th century, when Luigi Galvani found a connection between electrical voltage and muscle reflexes in frogs. It happened almost simultaneously with Alessandro Volta's discovery of the battery (Piccolino, 1997). A more profound understanding of the electrophysiological properties of individual cells came significantly later though.

An important landmark was the discovery of the *action potential* by Emil du Bois-Reymond, which, however, has first precisely been described by Julius Bernstein in 1868 (Bernstein, 1868; Seyfarth, 2006). The action potential is characterized as a transient change in the voltage across the cell membrane and it later turned out to be essential for information transmission in neural networks. To understand the underlying electrochemical processes, it is vital to consider the resting potential first.

Resting potential

Almost all animal and plants cells have an electrical potential in the soma that is different from outside the cell body. This resting potential is maintained by the lipid bilayer of the cell membrane which acts as an electrical insulator, preventing electrical current to flow directly through it. Instead, a large variety of proteinic ion channels are embedded into the cell membrane and allow certain types of ions to flow from the cytosol into the extracellular space more easily than others. In particular, the neuron membrane is more permeable for potassium than, for example, for sodium or calcium. In effect, the concentration gradient between the two membrane sides equilibrates for potassium until the electrical potential that is mostly created by the left-behind charges of the other ion types balances the outflux of potassium. This electrochemical equilibrium results in a static potential of the cytosol with respect to the extracellular space. Its amount depends on the various channel concentrations and types within the membrane (Alberts et al., 2007; Kandel et al., 2013). In addition to this passive ion diffusion, the resting potential is also influenced by proteins that actively pump ions through the membrane under consumption of adenosine triphosphate (ATP). Most important here is the Na^+-K^+-pump which regulatorily counteracts the effect of passive Na^+ and K^+ diffusions. In neurons, the sum of these mechanisms amounts to a voltage of around $-70\,\mathrm{mV}$ with respect to the extracellular space. The Nernst equation for nerve cells from which this potential

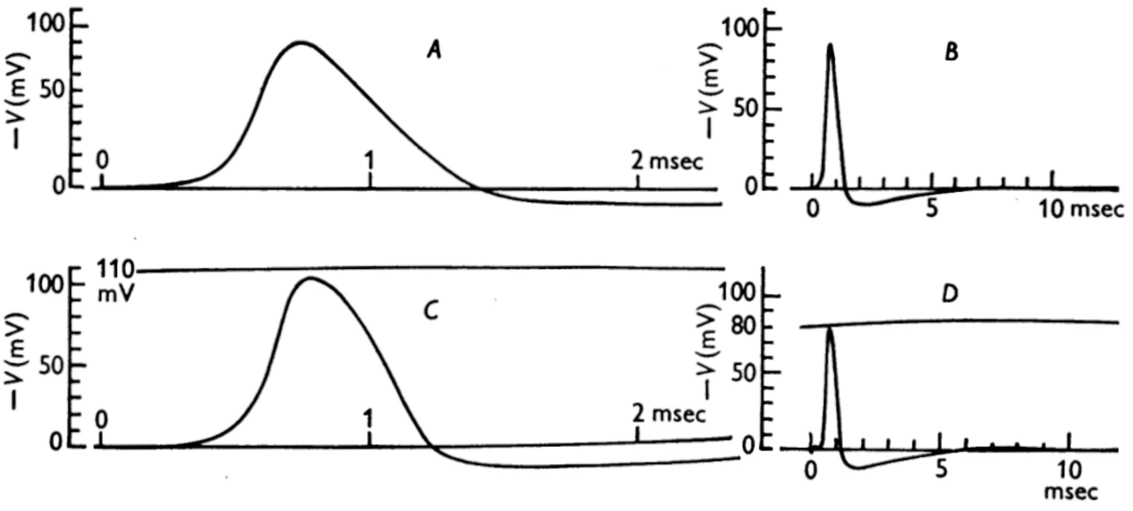


Figure 2.3: Real and calculated action potentials. A/B: Solution of the Hodgkin-Huxley equations plotted at two different time scales. C/D: Recorded action potential from a squid axon. Taken from the original publication by Hodgkin and Huxley (Hodgkin and Huxley, 1952a). Reprinted with permission from John Wiley and Sons and Copyright Clearance Center. Copyright 1952 The Physiological Society.

can be calculated was also first derived by Bernstein (1902) and later extended and generalized by Goldman (1943).

Action potential

However, the membrane potential V_{mem} of a nerve cell is not only found stable but can also be influenced by external stimuli like current injections. Such influences take approximately linear effect on V_{mem} over a range of a few mV. I.e., the membrane voltage response is in proportion to the amount of the applied stimulus. Around a certain threshold voltage V_{th}, however, a nerve cell quickly depolarizes (on the order of 1 ms) and increases V_{mem} overproportionally. After reaching a maximum voltage of about 30 mV to 40 mV, V_{mem} quickly decreases again, *hyperpolarizing* to a value $V_{\mathrm{reset}} \approx -90\,\mathrm{mV}$ that is even lower than its resting potential.

The cause of this is another class of membrane proteins, *voltage-gated ion channels* whose permeability is influenced by the voltage difference across the membrane. In the hyperpolarization phase, the voltage increase is dominated by voltage-gated sodium channels that increase the influx of Na^+ ions from outside the cell, driven by voltage and concentration differences. Being sensitive to an increase in voltage, this mechanism creates a positive feedback on itself, increasing the Na^+ permeability as the voltage rises. At a certain voltage, however, this gets counteracted by mainly three effects. One is that sodium approaches its electrochemical equilibrium, reducing the speed at which ions flow through the open channels. The second effect is that the voltage-gated Na^+ channels become inactive again when the voltage gets too high. And the third effect is that at a certain voltage also voltage-gated K^+ channels become active, increasing the permeability of sodium again, thus, causing V_{mem} to

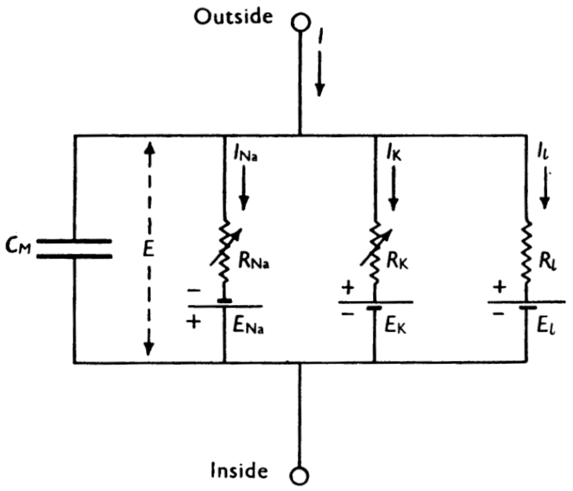


Figure 2.4: Circuit diagram of the Hodgkin-Huxley model taken from the original publication (Hodgkin and Huxley, 1952a). Reprinted with permission from John Wiley and Sons and Copyright Clearance Center. Copyright 1952 The Physiological Society.

reapproach its original resting potential.

The high permeability for sodium, though, leads to a *hyperpolarized* membrane state for a *refractory period* of around $\tau_{\text{ref}} \approx 2\,\text{ms}$, during which the neuron is unable to undergo the same chain of de- and hyperpolarization. This entire cycle is called action potential.

Its shape is very consistent over multiple repetitions. That is to say, the duration and the amplitude can be considered constant for all action potentials of a specific cell.

Hodgkin-Huxley model

The interplay of ion concentrations and permeabilities that has been described above was first quantitatively and comprehensively summarized by Alan Hodgkin and Andrew Huxley in an extensive series of papers in 1952 (Hodgkin and Huxley, 1952c,b,d,a). The two succeeded in formalizing a set of differential equations that can reproduce the kinetics of action potentials as observed in the giant axons of squids. The significance of these equations can hardly be overstated, as they provided the evidence, that the biochemical properties of the neuron, and in particular its membrane, have been understood to a degree that allowed quantitative prediction.

The model's equivalent circuit diagram is given in figure 2.4. Outside and Inside denote the extracellular space and the cytosol, respectively. C_M on the left side represents the membrane capacitance of the entire cell. R_{Na}, R_K are variable resistors that represent the inverse conductances of the sodium and potassium channels. Each is connected to the outside of the cell compartment and to one of the voltage sources, E_{Na} and E_K, which implement the Nernst potentials of the respective ion type. In addition to the two ion channels, a third series of a voltage source and a resistor is connected in parallel. E_l is the resting or *leak* potential of the neuron, i.e., the net potential that is achieved by all electrochemical balances and ion pumps in the resting state. R_l is the *leak resistance* that connects the cytosol to this potential. C_M (identical to C_{mem} in later sections) and R_l together define the membrane time constant τ_{mem}, i.e., the relaxation time with which the membrane decays back to its resting state, when both currents I_{Na} and I_K are negligible. Although the circuit seems to consist only of linear components, the resistances R_{Na} and R_K depend nonlinearly on the voltage E across the membrane (V_{mem} in later sections). Note that, the circuit in figure 2.4 does not explicitly account for any external stimuli.

Figure 2.3 shows two action potentials at two different time scales, each. The top row (A and B) shows the action potential as calculated from the Hodgkin-Huxley equations. The bottom row (C and D) shows actual recordings from a squid axon again for two different time scales. The difference in amplitude in D is explained by the fact that the axon had already been used for several hours before the recording. The model nicely reproduces the time scales, amplitudes, and the shapes of the biological standard. However, most of the terms, in particular the sodium and potassium currents and their nonlinear modulation, are mostly relevant to producing the action potential. If the membrane is below V_{th}, these term have negligible behavior on the kinetics by but bloat the computational complexity of the model. Especially for simulations that comprise networks of multiple cells, it is therefore desirable to find simpler neuron models.

FitzHugh – Nagumo model

Many of these are based on substituting the various chemical potentials by a single potential that represents the membrane voltage. A notable among them is the FitzHugh – Nagumo model (FitzHugh, 1961; Nagumo et al., 1962), which replaces the set of four highly nonlinear differential equations of the Hodgkin-Huxley model by two much more compact differential equations with a single remaining nonlinearity. Still, it is able to reproduce a vast range of observable cell phenomena (Izhikevich and FitzHugh, 2006).

Another interesting aspect about it is that the second eponymous author did not actually contribute to deriving the equations but presented an electronic circuit that emulated them. The intended purpose of this circuit was to overcome the attenuation and distortion that signals suffer from when traveling through (back then) conventional pulse transmission lines. The authors speculated that "an electric pulse signal which is transmitted along an animal nerve axon suffers neither attenuation nor distortion, regardless of the distance covered". Their intention was "to realize such a pulse transmission line by simulating the animal nerve axon". (Nagumo et al., 1962). This biomimetic approach to solving a technical problem that existed back then is worth being contrasted with the approach of contemporary neuromorphic engineering (section 2.2). The latter is ultimately concerned with overcoming current technical limitations in large scale brain simulations by mimicing the architecture of the brain itself.

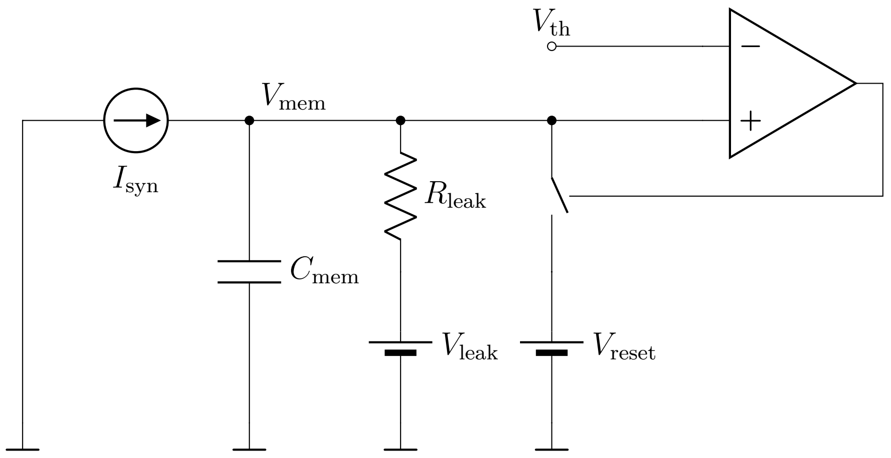


Figure 2.5: Equivalent circuit diagram of the LIF equations (equations (2.1) and (2.2)). A synaptic input current I_{syn} flows on the membrane capacitor C_{mem} that holds the voltage V_{mem}. Moreover, a leak resistor R_{leak} connects the leak potential V_{leak} to this node. If V_{mem} becomes greater than V_{th}, the comparator triggers a switch that connects V_{mem} to the reset potential V_{reset}. Thereafter, $V_{\text{mem}} = V_{\text{reset}} < V_{\text{th}}$, thus, opening the switch again.

Leaky integrate-and-fire model

An even further simplified mathematical neuron description is given by the leaky integrate-and-fire (LIF) model (Gerstner et al., 2014; Petrovici, 2016). This model entirely avoids nonlinearities in the differential equations by modeling the neural dynamics without accounting for action potentials. Instead, the action potential is substituted by a binary event that takes non-continuous action on the membrane potential for a defined amount of time. Interestingly, the model was first proposed by Louis Lapicque in 1907; long before Hodgkin's and Huxley's work (Lapicque, 1907; Abbott, 1999). Lapicque's model is not a set of equations derived from underlying physical principles but a heuristic and successful attempt to model the behavior he observed in real neurons phenomenologically.

The model consists of a continuous linear differential equation (equation (2.1)) and a conditional noncontinuous operation (equation (2.2)):

$$\tau_{\text{mem}} \cdot \frac{dV_{\text{mem}}}{dt} = (V_{\text{leak}} - V_{\text{mem}}) + R_{\text{leak}} \cdot I_{\text{syn}} \tag{2.1}$$

$$V_{\text{mem}} \leftarrow V_{\text{reset}} \quad \text{if} \quad V_{\text{mem}} > V_{\text{th}} \tag{2.2}$$

Here, V_{mem}, V_{leak}, V_{reset}, and V_{th} are the membrane, leak, reset and threshold potential, respectively. τ_{mem} is the membrane time constant and R_{leak} the leak resistance. Moreover, a synaptic input current I_{syn} is connected to the membrane capacitor. Figure 2.5 shows the equivalent circuit schematic of these equations. As long as V_{mem} stays below V_{th}, C_{mem} and R_{leak} act as a low pass filter on the synaptic input signals. The time constant of this filter is equivalent to the membrane time

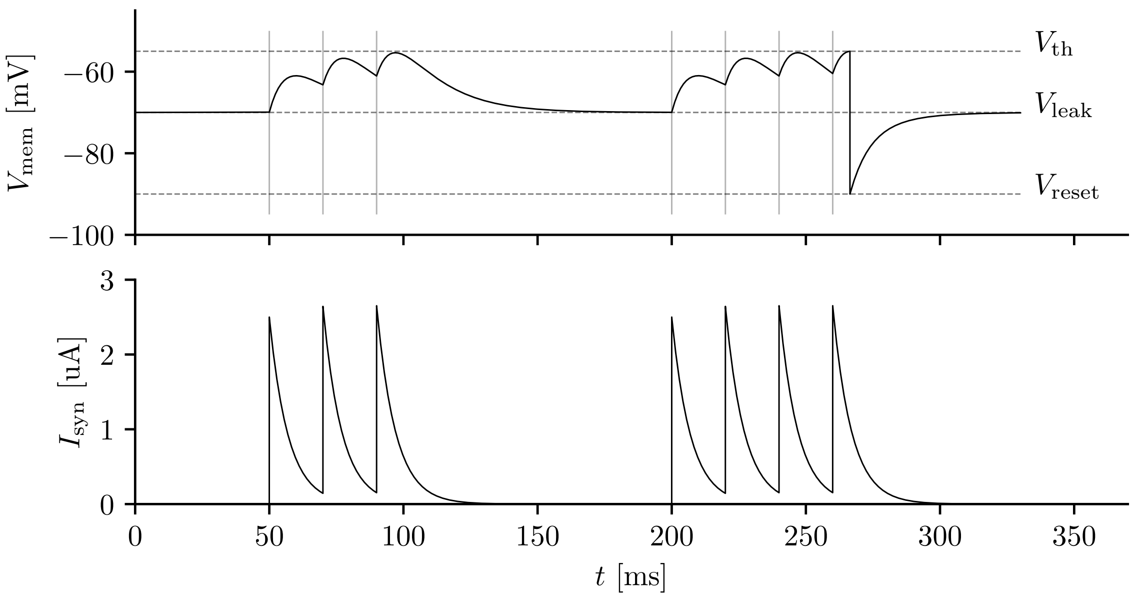


Figure 2.6: Bottom: Constructed synaptic input current, based on a superposition of immediately rising and exponentially decaying current pulses. Top: Response of the LIF equations (equations (2.1) and (2.2)) to this stimulus. The dashed horizontal lines indicate the three potentials V_{th}, V_{leak}, V_{reset}, and the faint vertical lines the onset of the current pulses from the lower plot. The parameters are $V_{\mathrm{leak}} = -70\,\mathrm{mV}$, $V_{\mathrm{th}} = -55\,\mathrm{mV}$, $V_{\mathrm{reset}} = -90\,\mathrm{mV}$, $R_{\mathrm{leak}} = 1\,\mathrm{M\Omega}$, and $\tau_{\mathrm{mem}} = 15\,\mathrm{ms}$.

constant $\tau_{\mathrm{mem}} = R_{\mathrm{leak}} \cdot C_{\mathrm{mem}}$. Whenever no input is present, i.e. $I_{\mathrm{syn}} = 0$, the membrane potential decays exponentially back to the leak potential V_{leak}. However, as soon as V_{mem} surpasses V_{th}, V_{mem} gets connected to V_{reset} via zero impedance, thus, resetting the membrane state.

The resulting kinetics are illustrated in figure 2.6. The lower plot shows a constructed synaptic input current, based on a superposition of immediately rising and exponentially decaying current pulses. The top plot shows the response of equations (2.1) and (2.2) to this stimulus. Between $t = 50\,\mathrm{ms}$ and $t = 150\,\mathrm{ms}$, three positive current pulses arrive on the membrane and cause the membrane potential to almost reach V_{th}. Between $t = 150\,\mathrm{ms}$ and $t = 200\,\mathrm{ms}$ no stimulus occurs and the membrane voltage decays back to V_{leak}. At $t = 200\,\mathrm{ms}$ another series of pulses starts that is sufficient to force V_{mem} over the threshold voltage and therefore to cause an action potential at $t \approx 270\,\mathrm{ms}$. Of course, since nothing in equations (2.1) and (2.2) accounts for it, the characteristic spike of the depolarization phase is missing (compare to figure 2.3). Moreover, the refractory period τ_{ref} is zero, so the LIF neuron from above would be able to fire again immediately after undergoing an action potential.

A simple way to introduce a refractory period and thereby enhance the equations' biological plausibility is to replace the auxiliary reset mechanism (equation (2.2)) by the following.

$$t_{\mathrm{spike}} \leftarrow t \quad \text{if} \quad V_{\mathrm{mem}} > V_{\mathrm{th}} \tag{2.3}$$

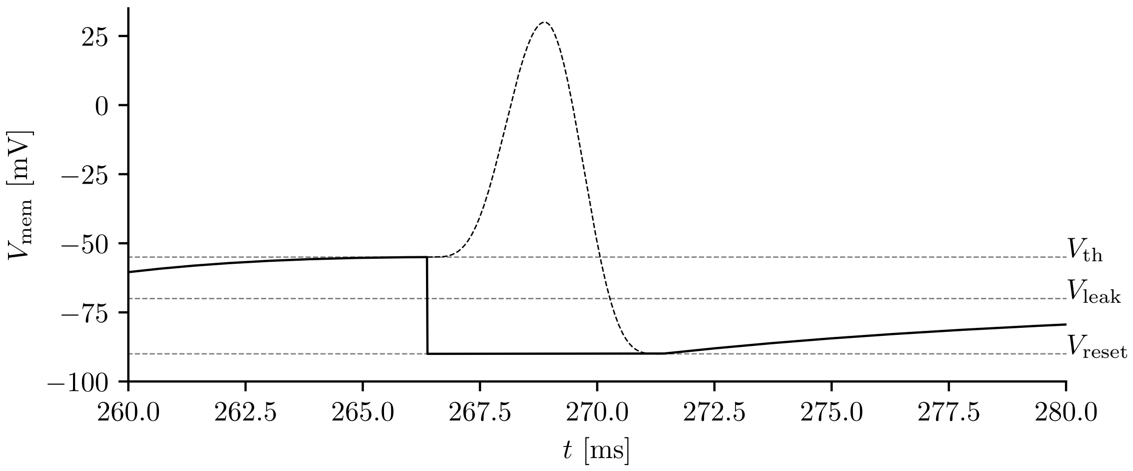


Figure 2.7: Response of the LIF equations with a non-zero refractory period (equations (2.1), (2.3) and (2.4)). The stimulus and the parameters are the same as in figure 2.6, except for $\tau_{\mathrm{ref}} = 3\,\mathrm{ms}$. The dashed action potential is artificially ploted on top and not related to the equations. It only indicates how the depolarization phase would fit into the refractory period.

$$V_{\mathrm{mem}} = V_{\mathrm{reset}} \quad \text{if} \quad t \in [t_{\mathrm{spike}}, t_{\mathrm{spike}} + \tau_{\mathrm{ref}}[\tag{2.4}$$

Besides resetting the membrane voltage at the moment it crosses V_{th}, equations (2.3) and (2.4) ensure that the reset voltage is maintained for the period τ_{ref}, effectively disabling the neuron for this time.

Figure 2.7 shows the response of equations (2.1), (2.3) and (2.4) when the same stimulus as in figure 2.6 is being applied. The zoom goes into a time interval around the action potential, where the neuron is refractory from approximately 266.4 ms to 271.4 ms. The dashed curve qualitatively indicates how the depolarization phase would fit into this period but is not related to the equations nor of any importance for the model.

Since biological neurons are assumed to be insensitive for stimuli during an action potential, the voltage trajectory of the membrane during that time is irrelevant in terms of neural information processing. In this context, it only matters that the neuron becomes susceptible again in the same state as it would be in after a precise action potential happened, i.e., at $V_{\mathrm{mem}} = V_{\mathrm{reset}}$.

When a LIF neuron is exposed to a persistent current stimulus, it responds with a regular sequence of spikes, because the threshold value is reached again and again after each reset. Figure 2.8 illustrates this behavior for different values of the input current. Assigning a spike rate to this regular sequence results in a transfer function as shown in the right part of the figure. Starting from $I_{\mathrm{syn}} = 0$, the output rate suddenly jumps to $r_{\mathrm{out}} \approx 20\,\mathrm{Hz}$, when the input stimulus is large enough to trigger a spike at $I_{\mathrm{syn}} \approx 1\,\mu\mathrm{A}$. From there on, it smoothly increases with I_{syn} and slowly approaches a maximum rate that is ultimately limited by the inverse refractory period $r_{\mathrm{max}} = 1/\tau_{\mathrm{ref}} \approx 333\,\mathrm{Hz}$.

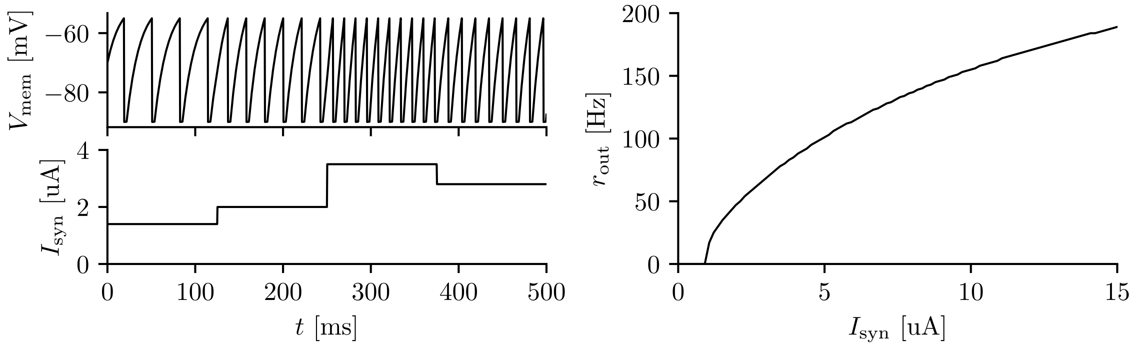


Figure 2.8: Rate response of an LIF neuron to a constant stimulus. Left: The synaptic current stimulus (bottom) is stepwise increased and decreased. The neuron (top) responds with a varying spike rate. Right: Output rate r_{out} of this neuron versus input stimulus.

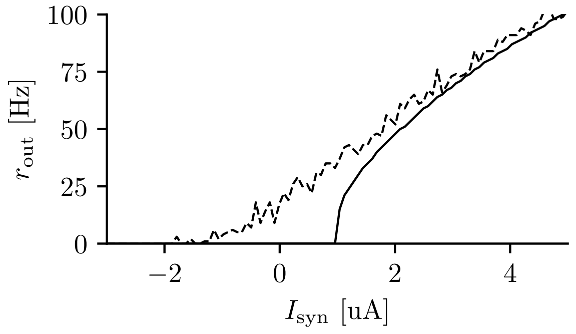


Figure 2.9: Rate response of an LIF neuron with (dashed) and without (solid) noise on the synaptic input current. The sudden increase at $I_{\mathrm{syn}} \approx 1\,\mu\mathrm{A}$ is smoothed out by the addition of noise. The parameters are the same as in figure 2.8.

The jump at the onset of regular spiking can be smoothed out by adding noise to I_{syn} which is illustrated in figure 2.9. Due to the resulting random fluctuations on its membrane potential, the neuron occasionally spikes, even when the average input current is below the spiking threshold. Because of the large number of chaotically firing presynaptic partners in biological neurons, a stochastic fluctuation on the total input current is usually the case. Therefore, a transmission curve like in figure 2.9 is more likely to be expected in cells that are embedded in a network than the smooth response function in figure 2.8.

If the firing rate is more important for information processing in a neural network than single spike events, the network is said to operate in the *rate limit*.

The internal dynamics of neurons lie typically within 1 ms to 100 ms. For example, cerebellar Purkinje cells in mice can reach a maximum firing rate of 625 Hz (Hurlock et al., 2008), while oxytocin neurons in rats fire only at a rate of $\mathcal{O}(1\,\mathrm{Hz})$ (Wang and Hatton, 2004). Most neurons, however, spike at an average rate of 10 Hz and a maximum rate of 100 Hz.

Overall, the LIF model is computationally efficient and reproduces three fundamental properties of biological neurons: it has a stable resting potential, it integrates over the synaptic inputs, and it mimics an abstract action potential in terms of creating a spike event and being refractory for a finite period.

However, it is not able to reproduce some phenomena observed in biological neurons, like for example, adaptation and bursting, without additional terms.

LIF extensions

Except for the spiking mechanism and the synaptic input, the circuit diagram in figure 2.5 is equivalent to the I_l-branch of the Hodgkin-Huxley model (figure 2.4). By adding the I_{Na} and I_K branches and set V_{th} above the peak voltage during the depolarization phase, the LIF circuit could therefore be turned into the Hodgkin-Huxley circuit without modifying its core structure. This shows that the linear structure of the LIF equations, as well as of the corresponding circuit, is easily extensible and modular.

So far, a large variety of extensions and modifications to the original LIF equations have been proposed. The most notable among them are probably the Izhikevich model (Izhikevich, 2003) and the adaptive exponential integrate-and-fire (AdEx) model (Brette and Gerstner, 2005).

Both of these models have a term that reduces or enhances the excitability (i.e. how much stimulus is required to produce an action potential) at a time scale of multiple action potentials. In effect, the firing rate of a neuron at constant stimulus is not constant like for the LIF neuron but it adapts by increasing or reducing its rate to the stimulus.

Moreover, both models have a nonlinear term that affects the membrane potential when it approaches the spike threshold. For the Izhikevich model, this is a quadratic nonlinearity that injects an additional current onto the membrane capacitor when V_{mem} is close to V_{th}.

For the AdEx model, this additional current is exponentially dependent on the differences between V_{mem} and V_{th}. The smaller the difference, the larger the current. Biologically, the nonlinearities in both models resemble the effect of the voltage gated ion channels in real cells.

Axonic spike

The point neuron descriptions from above implicitly neglect spatial effects, as they only include ion concentrations and electrical potentials in a scalar manner. In actual neurons, both are potentially diverse throughout the entire cell volume. To understand how an action potential spatially affects a neuron, a cell can be considered whose entire cytosol is first in a static resting state, i.e. $V_{\text{mem}} = V_{\text{leak}}$. If one point within the cytosol close to the membrane is then stimulated sufficiently enough to trigger an action potential, the resulting voltage increase will affect the neighboring membrane areas as well, causing them to also enter the depolarization phase. In effect, a wave is created that spreads radially from the initial point. As the membrane becomes refractory after it experienced an action potential, the wave leaves inactive membrane areas behind and thus, only travels away from the initial excitation and not towards it. Most importantly, if the soma experiences an action potential, the wave necessarily arrives at the axon hillock and finally at the axon which it then travels along. Since almost all action potentials originate from the dendritic tree or from the soma, the action potential travels practically always from the soma to the end of the axon. When this spike arrives at the end of the axon, it causes a biochemical reaction that releases neurotransmitters from the presynaptic cell into the synaptic cleft.

2.1.3 Dynamics of synapses

The released neurotransmitters then bind to receptors in the membrane of the postsynaptic cell, which results in an influx or outflux of ions and, hence, in a change in the membrane voltage. In real cells, this postsynaptic potential (PSP) is both a spatial and a temporal effect that is often described by the cable equations (Gerstner et al., 2014). In point neurons, only the temporal aspect is taken into account. Depending on the receptor and neurotransmitter type, the PSP can either be a positive or a negative voltage shift, but per synapse, it can only be one of both. The first type is called *excitatory* and the second type *inhibitory*.

The amplitude of a PSP mostly depends on the size of the respective synapse (Kandel et al., 2013). Regardless of the exact biological mechanisms, however, a modeled synapse can simply be assigned a weight w, that encodes both, the polarity and the amplitude of the PSP.

The effect of the ion stream caused by the neurotransmitters is often modeled as a current pulse of a certain length and shape onto the postsynaptic cytosol. Within this *current-based synapse model*, the total synaptic input of a neuron with index j can be defined as a series of weighted kernel functions:

$$I_{\mathrm{syn},j}(t) = \sum_i \sum_f w_{ij} \cdot \kappa(t - t^f_{\mathrm{sp},ij}) \tag{2.5}$$

Here, w_{ij} is the synaptic weight that connects the axon of neuron i to the dendrite or soma of neuron j. $t^f_{\mathrm{sp},ij}$ is the time when the f^{th} spike of neuron i arrives at neuron j. Hence, the sum goes over all presynapic neurons as well as over all spikes, each of them has sent so far. κ is the synaptic kernel function. In the previous illustration of the LIF equations in figure 2.6, the neuron received a series of kernel function of the shape

$$\kappa_{\exp}(t) = \Theta(t) \cdot e^{\frac{-t}{\tau_{\mathrm{syn}}}}. \tag{2.6}$$

This is known as an exponential kernel with the *synaptic time constant* τ_{syn}. Here, Θ denotes the Heaviside step function.

As equation (2.6) is also the Green' s function of the differential operator $\partial_t + \frac{1}{\tau_{\mathrm{syn}}}$, equation (2.5) can also be written as

$$\frac{d\,I_{\mathrm{syn},j}}{d\,t} = -\frac{1}{\tau_{\mathrm{syn}}} \cdot I_{\mathrm{syn},j} + \sum_i \sum_f w_{ij} \cdot \delta(t - t^f_{\mathrm{sp},ij}), \tag{2.7}$$

if $\kappa = \kappa_{\exp}$. The synaptic input current is therefore also representable as a first-order differential equation, which can be expressed in a linear equivalent schematic. There are further Green's functions which are suitable for use as synaptic kernels, but equation (2.6) covers all cases that are relevant within this document. Figure 2.10 shows a synaptic current as it can be constructed via equation (2.6) assuming various spikes arriving at various synapses.

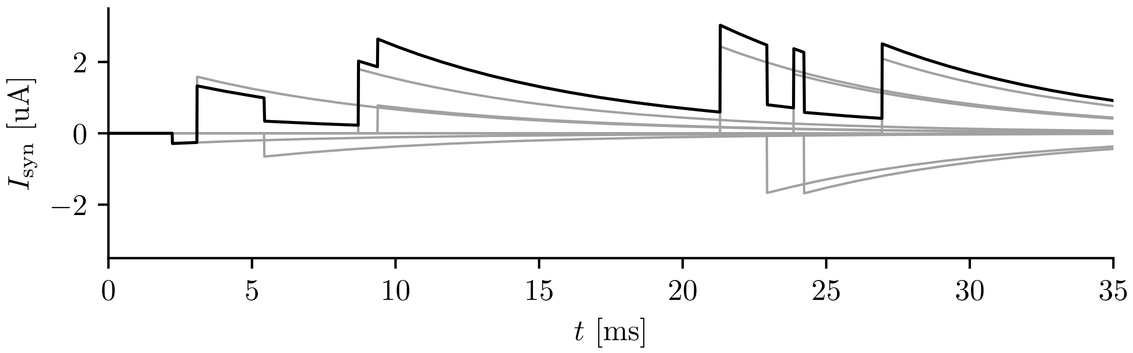


Figure 2.10: Synaptic input current as obtained from equation (2.7). The gray curves are the single PSPs $w_{ij} \cdot \kappa_{\text{exp}}(t - t^f_{\text{sp},ij})$ originating from various spikes $t^f_{\text{sp},ij}$ and synapses w_{ij} and the black curve is the assembled synaptic input current I_{syn}. Inhibitory synapses cause negative PSPs, excitatory synapses positive PSPs.

Short term plasticity

Similar to the adaptation in neurons that was briefly mentioned in section 2.1.2, also synapses can adapt to repeatedly occurring stimuli by becoming increasingly or decreasingly sensitive to incoming spikes. One of the most prominent models to replicate this behavior was proposed in Tsodyks and Markram (1997) and Tsodyks et al. (1998). The first work provides an analytical description of *short term depression* that assumes a successive depletion of neurotransmitters within the axon tip: If many spikes arrive in short time intervals, the available neurotransmitters get used up and the PSPs become weaker in amplitude in effect. When stimuli are absent, neurotransmitters can be regenerated and, thus, also the PSP amplitudes recover.

The second publication extends the mechanism by *short term facilitation*, i.e., a synapse produces successively larger PSPs when frequently triggered. Biologically, this is explained by an increased release probability of neurotransmitters, due to an influx of calcium into the axon after an action potential.

Although both phenomena happen on a time scale of multiple spikes, they can be considered as short term effects as compared to, for example, synaptic weight changes or synapse formation. Therefore, this type of synaptic modulation is called short term plasticity (STP).

Note that, STP is only dependent on the presynaptic spikes and therefore only on presynaptic properties. In contrast, other synaptic modulations also depend on postsynaptic variables, like the timing of the postsynaptic spikes.

Spike-timing dependent plasticity

One such mechanism has first been formalized by Donald Olding Hebb in *The Organization of Behavior*: "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased." (Hebb, 1949). This principle, known as *Hebbian*

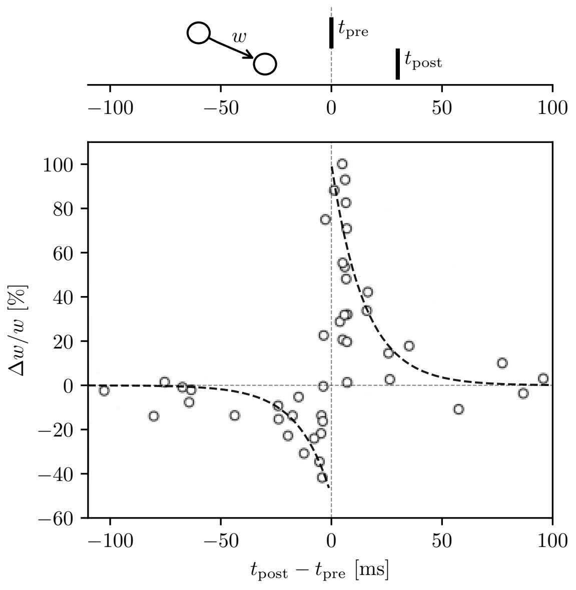


Figure 2.11: Relative weight change of a synaptic strength with respect to the difference between pre- and postsynaptic spike times. Data points are extracted from Bi and Poo (1998) and the accompanying drawing is inspired by Sjöström and Gerstner (2010).

learning, is often compressed into the slightly oversimplifying catchphrase "what fires together, wires together." What is meant by that is that the efficacy is increased in synapses whose presynapic neuron often cause the postsynaptic neuron to fire. Since the membrane potential of neurons intrinsically strives back to the resting potential, their memory of single spike events decays over time. Short time intervals between a pre- and the postsynaptic event should therefore indicate a higher causal relationship than long time intervals. Anticausal relations, i.e., when the postsynaptic neurons spikes before than the presynaptic one, should, on the other hand, have no or even a negative effect on the synaptic efficacy.

Firm experimental evidence for this type of spike-timing dependent plasticity (STDP) has first been provided by Bi and Poo (1998). If the individual weight changes are modeled as exponential functions of the spike time differences, according to Kempter et al. (1999), the mechanism can be expressed as

$$\Delta w = \sum_{t_{\text{pre}}} \sum_{t_{\text{post}}} W(t_{\text{post}} - t_{\text{pre}}) \tag{2.8}$$

with

$$W(x) = A_+ \cdot e^{-x/\tau_+} \quad \text{for } x > 0 \tag{2.9}$$

$$W(x) = A_- \cdot e^{x/\tau_-} \quad \text{for } x < 0. \tag{2.10}$$

Here, $W(x)$ is the effect of a single pair of pre- and postsynaptic spike events. $A_{+/-}$ is the amplitude of the causal/acausal weight change, and $\tau_{+/-}$ are the causal and acausal STDP time constants. t_{pre} and t_{post} denote the pre- and postsynaptic spike time, of course. The data from Bi and Poo (1998) together with two exponential functions for the causal and acausal branches, is shown in figure 2.11.

STDP is an important concept from a purely biological perspective as well as from the perspective of computational neuroscience, primarily because it constitutes an important building block for various biologically derived learning rules. A few relevant examples are given in Sjöström and Gerstner (2010); Frémaux et al. (2010); Neftci et al. (2014); Frémaux and Gerstner (2016).

2.1.4 Biological cybernetics

"Now that the concept of learning machines is applicable to those machines which we have made ourselves, it is also relevant to those living machines which we call animals, so that we have the possibility of throwing a new light on biological cybernetics."
– Norbert Wiener, Cybernetics, 1965

So far, a number of mathematical descriptions of cell physiological observations concerning neurons and synapses have been introduced. The following is about larger compounds of cells and their higher level organization and function in animals.

In contrast to plants, which are able to follow their autotrophic, sessile life cycles based on slow hormonal control, animals require faster signal transmission, since rapid movement is an essential prerequisite for foraging and for escaping predators (Wehner and Gehring, 2013). As nerve cells are capable of signal processing and transmission on time scales as short as a few milliseconds, they form the implementing substrate. While many primordial animals, such as jellyfish or starfish, tend to have diffuse networks of neurons, simple bilateral species such as flatworms already show higher concentrations of nerve cells in the head region where food is taken up into their bodies (Wehner and Gehring, 2013). The ability to analyze matter prior to ingestion of food using tactile, chemical, or optical sensors represents an obvious and significant evolutionary advantage. A comprehensive collection of sensors is not only useful for analytical purposes, but also enables efficient hunting due to the ability to track and trace. The Cambrian explosion, the greatest diversification event in the history of life some 541 million years ago, was, among other things, shaped by the emergence of quicker and stronger predators (Bengtson, 2002). There are even indications that this extreme diversification was supported by the emergence of eyesight, which enabled precise remote detection and spatially planned predation (Parker, 2003). The schematic plan of modern arthropod brains also dates back almost to this time (Ma et al., 2012). However, recent studies indicate a much earlier development of vision, which goes back at least to the last common ancestor of chordates and arthropods (Joly et al., 2016), i.e. about 620 million years ago (Edgecombe et al., 2011). In any case, from an evolutionary perspective, as well as from more abstract considerations about information processing, the development of nervous tissue in animals seems to be inseparably linked to the formation of sensors and muscles (Chiel and Beer, 1997).

In some species, particularly in the vertebrates, the size and complexity of neural networks has increased over the course of the evolution. The more complex these networks grew, the more complex the resulting capabilities and behaviors became. Higher level skills, like for example strategic hunting, or anticipation of predator behavior, led to higher level evolutionary advantages, which likely caused a positive feedback on the development of the responsible brain regions (Dukas, 2004). The species that is typically credited with the highest cognitive abilities on our planet are humans. Evidence that they in fact are is provided by a long list of remarkable achievements that other species did not yet achieve (as far as we know). Using and shaping complex language and metalanguage, sending robots on interplanetary journeys, or writing doctoral theses on neuromorphic hardware, to name few. Fascinatingly, all these capabilities so far remain somewhat opaque in the face of the brain's slippery, wet, not very aesthetic shape.

The human brain is taken here as a benchmark from which some dimensions and properties are derived for the considerations in the next sections. Note, however, that many properties are conserved among different vertebrate species. For example, the cerebellum of birds (and thus of dinosaurs) is composed of cells that are very similar to the cells in the human cerebellum (Arends and Zeigler, 1991). The same holds true for the cerebral cortex in mice and humans (Kandel et al., 2013). Even arthropods share synaptic classes and cell organizations with us (Smarandache-Wellmann, 2016). Therefore, some properties, like the time scales and sizes of cell-level phenomena can be considered similar for approximative modeling aspects. Other properties, like the brain size in terms of its weight or its number of neurons and synapses are, however, obviously divergent among species. As Williams and Herrup (1988) put it, "[t]he total number of neurons in the central nervous system ranges from under 300 for small free-living metazoans such as rotifers and nematodes [...], through about 30 million for the common octopus and small mammals such as shrews [...], to well over 200 billion for whales and elephants".

A 2009 study about the human brain found that it contains 86.06 ± 8.12 billion neurons of which approximately 69 billion are located in the cerebellum and 16 billion in the cerebral cortex (Azevedo et al., 2009). Pyramidal neurons in the cerebral cortex typically receive $2 - 30 \times 10^3$ synaptic inputs each, depending on the region (Elston et al., 2001; DeFelipe et al., 2002). The Purkinje cells of the cerebellum have a higher number of approximately 1.7×10^5 synapses per neuron (Napper and Harvey, 1988).

For the sake of simplicity, the following estimates therefore assume that the human brain contains roughly $\mathcal{O}(10^{11})$ neurons with $\mathcal{O}(10^4)$ synapses each.

2.2 Neuromorphic systems

"Living matter certainly has a fine structure more relevant to its function and multiplication than that of the parts of a nonliving machine, though this may not be equally the case for those newer machines which operate according to the principles of solid-state physics."
– Norbert Wiener, God & Golem, Inc., 1964

2.2.1 von Neumann architectures

To understand the motivation behind novel computer architectures, it necessary to briefly illustrate what conventionial architectures are already capable of.

The most widespread system design in contemporary computers goes back to a draft report by John von Neumann from 1945 on the EDVAC (Electronic Discrete Variable Automatic Computer) (Von Neumann, 1993). This document contains a description of a digital computer that consists of four essential components:

- An input device
- A central processing unit (CPU)
- A central memory that contains data and programming instructions
- An output device

Moreover, the CPU is divided into two subunits, a control unit that manages the programming flow and a processing unit that performs arithmetic logic operations. Both have working registers for programming instructions and data, respectively. While there are multiple variations and extensions to this concept, the core principle of a centralized computation unit with memory access is what is typically referred to as a von-Neumann-architecture.

Resembling the principles of a universal Turing machine, von-Neumann computing devices are limited in what they can compute only by the available memory and time resources. Since its development in the 1950's, qualitative advancement in digital computing was largely a consequence of quantitative increases of computation speed and memory due to improvements in silicon based semiconductor devices.

Computer programs are typically compiled into a list of hardware instructions that is then consecutively executed on a processor. Even machines that have a large number of parallelly distributed processing cores, like for example graphics processing units (GPUs), work by the same principle.

As outlined in the previous section, biological neural networks operate in a very different manner. One prominent difference is the intrinsically distributed and massively parallel arrangement of their information processing units, i.e., the neurons and synapses. Being independent of the computing progress of other cells, a neuron, unlike a CPU, implements its own function continuously and uninterruptedly. When new information arrives, it gets processed immediately.

Another outstanding difference is that memory and computation are combined and distributed over the entire neural network. The amplitude of a post-synaptic potential is computed by a synapse whose strength can abstractly be identified as a memory. Hence, the physical structure that implements the memory simultaneously performs the computation.

Mathematical neuron descriptions, like for example the LIF equations (equations (2.1) to (2.4)), can of course be solved numerically on von Neumann machines. By introducing discrete time and sequentially iterating over the involved synapses and neurons, a conventional computer can simulate a large scale neural network to an extent that is almost exclusively limited by the available memory and computation time. However, the latter is what currently predominantly prevents conventional computers from scaling up the size of simulated neural networks, to an arbitrary extent.

2.2.2 Computational complexity of brains

Since, as of today, no comprehensive fundamental theory of computation in biological neural networks has been found, deriving a reliable estimate of the true computational complexity of brains is a highly speculative endeavor. One way to do so, however, is to assume that each synaptic event corresponds to one floating point operation.

Following this reasoning, the operations that are associated with a neuron's action potential occur much less frequent than operations associated with postsynaptic potentials, because the number of synapses is 10^4 times larger than the number of neurons. For an estimate of the computational scale, the action potentials are therefore negligible in comparison. Assuming further an average neural spike frequency of 10 Hz, the total number of floating point operations per second (flop s^{-1}) is $10^4 \cdot 10^{11} \cdot 10\,\mathrm{Hz} = 10^{16}$ flop s^{-1}. If the resolution of the synaptic weights is only 1 byte, we further require $10^{11} \cdot 10^4 \cdot 1\,\mathrm{B} = 1\,\mathrm{PB}$ of memory for the connectivity matrix. In order to compute the synaptic action, every entry of this matrix has to be accessible at an average spike rate of 10 Hz. Hence, the data rate for accessing the synaptic weights alone is approximately $10\,\mathrm{PB\,s^{-1}}$.

So far, the estimate does not contain synaptic mechanisms like STP, or STDP. If these ought to be included, another set of roughly $10^{11} \cdot 10^4$ differential equations have to be solved and must be iterated over at a rate not less than the average spiking rate. Thus, at least another $10\,\mathrm{Pflop\,s^{-1}}$ would be required.

Note that, this estimate assumes point neurons, i.e., that all ion concentrations can be abstracted into a scalar membrane potential. It also neglects numerous existing physiological phenomena, like synaptic delays, extra-neural dynamics (like potentials in glia cells, or dopamine concentrations, etc.), or spatial effects like the dendritic structure of neurons. Considering these simplifications, the given estimate should be regarded as a lower bound of the number of computational operations that are *actually carried out* in the human brain.

The actual amount of computation that is necessary to implement the brain's function might be lower though, because of noise, redundancy, and non-optimal implementation in the biological substrate. As, so far, not enough is known about how spiking neural networks are compressible, arguing for a lower number would be even more speculative, however.

2.2.3 The von-Neumann brain

This section aims at evaluating the feasibility of a network composed of the estimated number of neurons and synapses in terms of a LIF neuron implementation on conventional computing architectures. It starts with some very crude and simplified estimates and then concentrates on existing hardware and software examples.

To achieve a biologically reasonable precision, a LIF neuron typically requires an integration time increment smaller or equal to $\Delta t = 0.1\,\mathrm{ms}$, and thus, 10 000 simulation updates per second (van Albada et al., 2018). On the other hand, every neuron receives input from 10 000 presynaptic partners each spiking at a rate of 10 Hz on average. The numerical operations to implement the synaptic inputs are therefore again dominant or at least equal compared to those for the neuron dynamics. For simplicity, we therefore continue with the above derived 10^{16} flop s^{-1} and $10\,\mathrm{PB\,s^{-1}}$.

According to the Green500 list, high performance computing (HPC) currently achieves an energy efficiency of up to around $20\,\mathrm{Gflop\,W^{-1}}$ (TOP500.org, 2020). If the above estimate is naively divided by this number, a power of 500 kW would be necessary to run a full scale brain simulation in real-time. Compared to a human brain, which consumes about 20 W on average, this is roughly four orders of magnitude less energy efficient. On the other hand, with a power requirement of a small to medium-sized factory, it is still within a feasible range. However, many factors are not included in this estimate.

First, the number of flop s^{-1} cannot be pumped out with a hundred percent efficiency. In between the actual floating point operations, data has to be transferred into and out of the various CPU registers, and external memory locations. Typically, without further optimization, 10 to 100 CPU cycles are easily waisted for a single floating point operation.

Second, as laid out above, the memory and hence, the memory bandwidth that the brain simulation requires, is far above what can be connected to the various CPU cores in terms of shared memory with the latency requirements that a plausible simulation demands. The memory access must therefore be put over further detours, wasting further resources. To get a more realistic estimate of the the actual power consumption of a brain simulation, one can take a look at published results.

Recently, for example van Albada et al. (2018) performed an optimized simulation of one cortical column (comprising approximately 80 000 neurons and 0.3 billion synapses, i.e. $\sim$0.0001 % of the entire brain) on a supercomputer. They obtained a minimal required energy of 5.8 µJ per synaptic event. If this value is linearly scaled up to a complete brain simulation, the power consumption already reaches 58 GW (the equivalent of 2 - 3 Three Gorges Dams[1]), which is roughly six orders of magnitude above the overly simplistic one-flop-per-synaptic-event estimate from above. Moreover, it still neglects the nonlinear increase in power demand due to the growing network complexity.

An existing class of computing devices that are better suited for parallel processing tasks are GPUs. Originally invented to efficiently solve computer graphic problems

[1]The maximum electric generating capacity of the Three Gorges Dam is reported to be 22 500 MW (Wikipedia, 2020b).

which typically involve matrix multiplications, and hence parallel arithmetic, they are very suitable for being reused for similar tasks in computational neuroscience and machine learning. In a another recent publication, Knight and Nowotny (2018) simulated the same model of a highly-connected cortical microcolumn (Potjans and Diesmann, 2014; van Albada et al., 2018) on a variety of GPUs. The authors showed that in this case GPUs perform better in terms of speed and power consumption than HPC and even some existing neuromorphic platforms. Among the best results they obtained was 0.3 µJ per synaptic event on a Nvidia Jetson TX2 device, which is roughly one order of magnitude below the HPC performance but still a stunning eight orders of magnitude away from what the brain requires for a synaptic operation, i.e. 2 fJ.

It is for this reason, that alternative computing approaches are being developed.

2.2.4 Parallel computation and memory

The result of the last section is that simulations of neural networks are intrinsically inefficient to implement on traditional architectures.

As indicated above, one of the greatest inefficiencies that manifest when simulating neural networks on von Neumann architectures is the serial *memory access* of massively parallel data. An approach to overcome this bottleneck is to locally combine memory and processing units. For example, if the synaptic weight is always and with low latency accessible to the unit that computes the postsynaptic potential, the overhead of loading the weight matrix disappears. Moreover, the network can be scaled up more easily because doubling the size of computing resources automatically doubles the required memory.

Another mentioned issue is that also the massively parallel *computation* per neuron and synapse has to be serialized on conventional architectures. Since a single microchip does not offer unlimited resources, this can be solved by trading the complexity of a single processing unit against the number of such units. GPUs for example, take this approach.

Both strategies can be pursued with both analog as well as digital computers. If the purpose of such a hardware device is exclusively the simulation or emulation of neural networks, then the device is commonly referred to as a neuromorphic system (Mead, 1990).

2.2.5 Neuromorphic devices

In the recent years even a number of large chip manufacturers engaged in building neuromorphic devices. For example IBM's TrueNorth (Merolla et al., 2014) or Intel's Loihi (Davies et al., 2018) are both complementary metal oxide semiconductor (CMOS)-based microchips that contain custom digital circuits to implement large clusters of neurons and synapses. Most of the neuromorphic projects, though, are still situated in academia (Furber et al., 2014; Benjamin et al., 2014; Qiao et al., 2015; Shen et al., 2016; Moradi et al., 2017; Schemmel et al., 2017; Neckar et al., 2018) or small university spin-offs, like aiCTX/SynSense, or brainchip. Neuron-inspired circuits have first been implemented in very-large-scale integration (VLSI), i.e. integrated into large scale semiconductor devices, in late 1980s and early 1990s (Mead, 1989, 1990; Douglas et al., 1995). Since then many different approaches for many different aspects of the involved problems have been taken. Most important, perhaps, is the distinction between digital and analog implementations, the second of which is mostly relevant for the current thesis.

The fundamental idea in analog neuromorphic engineering is to replace digital circuits that solve differential equations numerically by circuits that make analog voltages *behave* like the state variables of the differential equations. The main motivation behind this idea is energy efficiency.

By far the most transistors that are currently existing on earth are implemented as CMOS devices. The most characterizing feature of this technology is that it uses complementary and symmetrical pairs of metal – oxide – semiconductor-based field-effect transistors (MOSFETs) (Weste and Harris, 2015). In this transistor type, the gate is only capacitively coupled to the semiconductor substrate. The energy that is required to charge such a gate in order to make the substrate conducting is approximately

$$E = V^2 \cdot C, \tag{2.11}$$

where V is the voltage across the gate and C is its capacitance. To a certain degree, C can be approximated as a plate capacitor that is formed by the semiconductor on the one side and the conducting gate on the other side, separated by a thin layer of insulating oxide. Thus, C scales with the intersection area between the two galvanically decoupled sheets. For this reason, the energy efficiency of semiconductor technologies scales roughly quadratically with the inverse minimal gate length.

Moreover, E scales quadratically with the voltage V that is applied across the transistor gate. In digital circuits, V is almost always driven from its maximal to its minimal value and vice versa. This ensures, that the transistors are either fully open, facilitating high currents and therefore fast switching of subsequent gates, or fully closed, preventing leakage currents, in the transient operating states. Therefore, the energy efficiency of a digital CMOS device also decreases with the squared operating voltage.

On the other hand, if analog computation is performed on a capacitor, the minimal voltage changes are only limited by the signal-to-noise ratio and are usually significantly smaller than the supply voltage. Moreover, the transistors involved in analog computation are also not necessarily bound to the digital binary states. Instead, they can operate in arbitrary configurations, even in the *subthreshold region*, i.e., when the gate voltage is below the threshold voltage required to activate the source-drain channel. The drain current in this region is proportional to

$$I_{\mathrm{D}} \propto e^{\frac{V_{\mathrm{GS}} - V_{\mathrm{th}}}{c \cdot k \cdot T}} \tag{2.12}$$

where V_{GS} is the gate-source voltage, V_{th} the threshold voltage, c a transistor specific constant, k the Boltzmann constant, and T the absolute temperature of the device. Due to the exponential dependency on V_{GS}, the currents through the transistors can be as low as only a few nA of even pA. In effect, the reported energy per spike of systems operating in this subthreshold regime can go down to the order of a few pJ when running in biological real-time (Livi and Indiveri, 2009; Indiveri et al., 2011).

However, equation (2.12) also reveals a fundamental disadvantage of the subthreshold approach, i.e., the exponential temperature dependency. Without active and precise temperature control, these circuits' behavior is as influenced by ambient temperature drifts as it is by the gate voltages. This temperature dependency is much less pronounced in circuits that operate in the ohmic region (i.e., $V_{\mathrm{GS}} > V_{\mathrm{th}}$ and $V_{\mathrm{DS}} < V_{\mathrm{GS}} - V_{\mathrm{th}}$) or in the saturation region (i.e., $V_{\mathrm{GS}} > V_{\mathrm{th}}$ and $V_{\mathrm{DS}} \geq (V_{\mathrm{GS}} - V_{\mathrm{th}})$).[2] In both cases, mostly the temperature dependence of V_{th} is influential which is typically on the order of only $0.5\,\mathrm{mV\,K^{-1}}$ to $3\,\mathrm{mV\,K^{-1}}$ (Weste and Harris, 2015). Most analog integrated circuits therefore avoid the subthreshold region.

However, transistors operating above or at threshold voltage cannot straightforwardly achieve drain currents that are as low. By tendency, this leads to faster circuit dynamics, as the voltage changes across capacitors are directly proportional to the related currents. The easiest way to overcome this obstacle in neuromorphic engineering is to embrace it and design the circuits such that all time scales are scaled down by a constant linear factor, the *acceleration factor* or *speed-up*. If, for example, the membrane time constant τ_{mem} and the refractory period τ_{ref} of the LIF equations (equations (2.1), (2.3) and (2.4)) are scaled by an acceleration factor α, the relative neuron dynamics remain unchanged while only the absolute time scales are compressed. E.g., if the average spike rate of such a neuron is r_{max} before scaling, it becomes $r'_{\mathrm{max}} = \alpha \cdot r_{\mathrm{max}}$ after scaling.

The qualitative impact of such an acceleration manifests in positive and negative aspects. The greatest shortcoming is, perhaps, that an accelerated neuromorphic system is usually not compatible with biological systems. Obviously this is to be seen negative for biological or medical applications that aim at interfacing nervous tissue with electronic devices. Not quite so obviously, it generally complicates interfacing the accelerated neural networks with external hardware, like for example, robotic sensors, actuators, or even conventional computers. There are two main reasons for this:

First, the input/output data bandwidth increases linearly with the speed-up, while the tolerated latency decreases linearly with it. For neural networks this is typically more challenging than, for example, for radio signals, because of the inherent massively parallel data representation. This is not only challenging for the interface itself but also for the connected entities. For example, if a neuromorphic ear receives an audio signal from a computer, a maximum jitter of maybe $\sim$100 µs (1/10 kHz) should not be violated if it operates in biological real-time. For a neuromorphic ear, operating 1000 times faster, the maximum allowed jitter becomes $\sim$100 ns, which is equivalent to just a few hundred CPU cycles on a typical computer. So far, in the context of accelerated neuromorphic hardware, only around 8.5 µs minimum latency have been achieved between a neuromorphic system and a computer while requiring highly competent and nontrivial modifications even on real-time operating systems (Müller, 2015). Concerning the parallel bandwidth, a quantitative estimate for the

[2] V_{DS} is the voltage across the transistor's drain and source.

optic nerve is given later in section 4.2.2.

The second reason is that most machines or interfaces are designed for human interaction. While standard computer monitors can be used to create a virtual reality for larval zebrafish (Engert, 2013), a frame rate of 50 Hz appears pretty much static to a neuromorphic system that is operating at a 1000-fold speed-up. Reaction times of a few dozen milliseconds are considered fast for robotic actuators (McClintock et al., 2018) but translate to reaction times of a few dozen seconds for a system with such a speed-up. The reason for this lies not only in arbitrary or coincidental preferences of the human species for certain time scales, but is related to the energy scales that we live in. The average energy and mass densities in our environment characterize the typical time scales of mechanical processes (see equation (4.1)).

However, both the bandwidth and latency requirements of accelerated neuromorphic hardware do not interfere with physical limitations but are merely technical problems that can be solved by investing engineering resources. In terms of virtual environments, specialized low latency simulators can be developed and coupled closely to the neuromorphic hardware. The corresponding design effort for such an auxiliary component is most likely below the design effort that has to be put into developing the system itself. It should not be neglected though. In terms of real-world robotic applications, the same holds true. Acceleration at reasonable energy scales can potentially be achieved by either miniaturizing mechanical systems, or by switching from mechanical actuation to sound or light based systems.

Overall, the advantages of an accelerated neuromorphics likely outweigh its disadvantages. Most importantly, the speed-up increases the experiment throughput to an extent that unlocks entire fields of research. For example, the neuronal development of vision in cats takes approximately ten weeks to complete (Tootle and Friedlander, 1989). Studying neural phenomena on such timescales must doubtlessly be one of the goals of large scale brain simulations, since all large brains in biology are developing over time scales of months or even years to decades. Apart from brain development, numerous other interesting learning phenomena, such as structural plasticity, unfold over several days to months (Grutzendler et al., 2002). Even these time scales are so significant for real-time simulators that statistical analyses cannot be carried out straightforwardly. With an acceleration of 1000, however, a day can be compressed to one and a half minutes, a month to 43 minutes, ten years to three and a half days. In this manner, the statistical capacity is amply sufficient to apply strategies such as hyperparameter optimization or systematic parameter sweeps even to large-scale emulations that study long-term phenomena.

One system that aims for the illustrated capabilities is the BrainScaleS system developed and operated at the University of Heidelberg.

2.2.6 BrainScaleS

The BrainScaleS system is an accelerated analog neuromorphic computer with a digital communication infrastructure. I.e, the neurons and synapses on BrainScaleS are emulated by power efficient analog CMOS circuits, while the spikes are being transported digitally. Like its name implies, the system is designed to scale to a virtually arbitrary size while maintaining a constant speed-up factor.

The first large scale version of BrainScaleS is centered around the so-called high input count analog neural network (HICANN) chip that contains 512 analog neurons with 224 synapses each that are accelerated by a factor of 10 000. The HICANNs are, like most microchips, produced on disk-like slices of crystalline silicon, called wafers. For technical reasons, the size of a continuous circuit layout cannot correspond to that of the entire wafer, but is instead confined to a rectangle with a side length of a few centimeters. These rectangles, also called reticles, define the largest producible unit on a wafer and are tiled next to each other to maximally exploit the available area resources. Within one reticle, eight HICANN chips find place and exchange spikes with each other by means of a custom communication infrastructure. Due to the mentioned technical limitations, each reticle, is isolated after wafer production. They can be joined together, though, by adding additional metal layers on top of to the wafer in a custom post-processing step. Ultimately, 48 reticles with 8 HICANNs each can be combined by this *wafer-scale integration* to obtain a system of 196 608 neurons and 44 040 192 synapses. Moreover, multiple wafer-scale systems can be interfaced with each other to further scale up the system size. Extensive information on the BrainScaleS system can be found in Schemmel et al. (2008, 2010); Brüderle et al. (2011).

Development of the HICANN chip went on in parallel to the development of the wafer-scale system. On the one hand, in terms of incremental improvements of the existing chip, on the other hand, in terms of developing an entirely new version. This second version, BrainScaleS-2, is manufactured by TSMC in a 65 nm process instead of the 180 nm node the previous chip was designed for. Apart form this transition to another process technology, a large number of circuits and system-aspects were added and improved in BrainScaleS-2. The next section starts with an overview of the system architecture.

3 BrainScaleS-2

"What I cannot create, I do not understand."
– Richard Feynman

The central component of the BrainScaleS-2 family is the analog neural network core (ANNCORE) which contains the neuron and synapse arrays as well as other supporting analog and mixed-signal circuits. A simplified overview of its architecture is given in figure 3.1. Digital spike events arriving in the ANNCORE are first received by the synapse drivers (vertically stacked triangles at the left of the synapse array in figure 3.1) which preprocess and then drive them across horizontal rows of synapses. In case a synapse receives a spike, it injects an analog current pulse onto a vertical synaptic input line stretching from the topmost synapse to the neuron at the bottom. These pulses are then collected and processed by the neurons which in turn may transmit spike events via outgoing digital lines. Each synapse further stores analog voltages reflecting spike correlations which can be used for STDP-like learning rules (see section 2.1.3). These signals and the membrane potentials of the neurons can be digitized by a parallel ADC, the CADC, at the top edge of the synapse array.

The ANNCORE is surrounded by a digital back end of control and communication subsystems as well as a digital coprocessor with a PowerPC instruction set (Wikipedia, 2020a). The main purpose of this coprocessor is to provide a versatile and efficient way to implement arbitrary learning and control algorithms. Since most of these take actions on the synaptic weights, the processor is referred to as the plasticity processing unit (PPU) (Friedmann, 2013). Its standard PowerPC instruction set is extended by a custom single instruction multiple data (SIMD) vector unit that can perform instructions on slices of 16 data entries at once which is particularly useful when operations on the synaptic weights are performed. The PPU is perhaps the most significant innovation over BrainScaleS-1 and gives BrainScaleS-2 its extraordinary versatility in terms of its applications.

The development of BrainScaleS-2 started with a series of smaller prototype chips (1.7×1.7 mm^2) that extended over three generations. Due to the powerful digital extensions, these chips are also called high input count analog neural network with digital learning system (HICANN-DLS). To indicate the respective prototype version, v2 or v3 is sometimes appended to this acronym. After the prototypes became mature, the development continued with a series of two full scale versions (4×8 mm^2), called high input count analog neural network - X (HICANN-X). Since the experiments presented in chapters 4 and 5 were conceived and refined before the fully sized system was available, they were carried out on one of the small versions, in particular on HICANN-DLSv2 (Aamir et al., 2016). The following describes some of the most important system components in more detail.

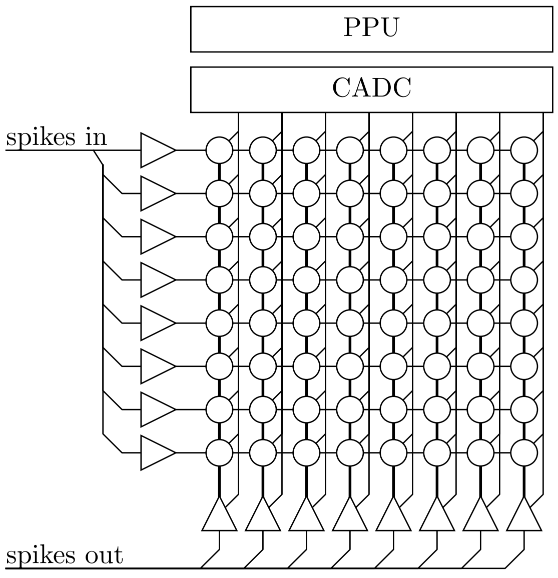


Figure 3.1: Simplified ANNCORE architecture and PPU. The synapse drivers receive incoming spikes events and buffer them horizontally into the synapse array. If a synapse receives a spike, it injects a current onto a vertical synaptic input line stretching from the topmost synapse to the neuron at the bottom. The neuron in turn may send a spike to outgoing lines. The correlation measurements obtained in each synapse and the membrane potential of the neuron can be connected to the inputs of the CADC at the top edge of the synapse array. The digitized results can then be read out and processed by the PPU.

Contribution

is a very complex project in which numerous people are and were involved explicitly or implicitly. In order to avoid listing them all, it may be assumed in most of this chapter that the contributions are not from the author, unless it is explicitly mentioned in the text. This rule is inverted for sections 3.2, 3.4 and 3.5, where the author was primarily responsible for the content. In these chapters it can be assumed that all contributions can be attributed to the author, unless it is explicitly mentioned in the text. For clarity, the most important contributions are also listed here:

The author designed the prototype baseboard and the HICANN-DLSv3 chip carrier board which are outlined in section 3.2. Both are based on previous designs by Matthias Hock. The author further designed the xBoard and the HICANN-X chip carrier, as outlined in section 3.4. In this case no previous design was adapted. Moreover, he designed the CADC of HICANN-DLSv3 and HICANN-X as outlined in section 3.5, with the exception of an operational amplifier by HBP collaboration partners from the Sabancı Üniversitesi in Istanbul and another operational amplifier from Matthias Hock (Hock, 2014). Both is mentioned again at the text passage where these components appear. Besides these three sections, the author contributed to some parts of the readout chain that was introduce in HICANN-DLSv3 and in form of some invisible labor related to the tape-outs of HICANN-DLSv3 and HICANN-X. Both is not explicitly mentioned in the text.

3.1 The prototype chips

In the prototypes, the ANNCORE comprises 32 neurons and 32 × 32 synapses. Moreover, it contains a 64 channel ADC, an analog parameter storage, the synapse drivers, and in v3 the digital neuron backend. The order of the descriptions of these subcomponents tries to follow the data flow of the spikes events outlined above and in figure 3.1.

3.1.1 Synapse drivers and synapses

Synaptic events

The main purpose of the synapse drivers is to forward the digital spike events horizontally into the synapse array. In particular, they control the synapses timing by means of a short digital voltage pulse. Furthermore, each spike event is represented by a 6 bit address, which identifies its source and each synapse stores a corresponding 6 bit value within a local static random-access memory (SRAM). If the label of an incoming event matches the locally stored one, a synaptic current digital-to-analog converter (DAC) injects a current pulse onto one of two vertical synaptic input lines which lead to the respective neuron. Depending on which line it is injected on, the synaptic pulse acts either excitatorily or inhibitorily on the neuron. The corresponding configuration is also done row by row by the synapse drivers. The amplitude of the current pulse is defined by a six bit *weight* which is also stored in a dedicated SRAM per synapse. The pulse width is derived from the above mentioned digital pulse coming from the synapse drivers and on the order of a few nanoseconds. Hence, the weight and the pulse length linearly influence the amount of charge injected into the synaptic input line.

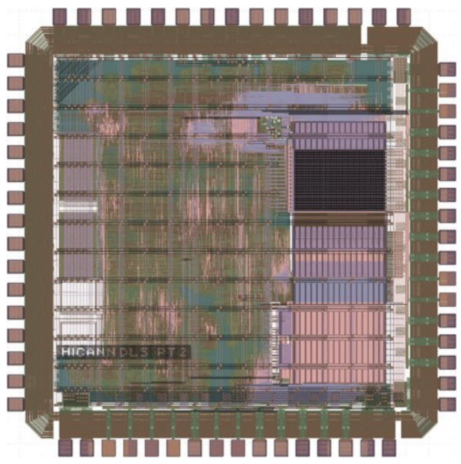


Figure 3.2: HICANN-DLSv2 layout. The chip measures $1.7 \times 1.7\ \mathrm{mm}^2$.

Short-term plasticity

While the pulse width can only be configured globally on HICANN-DLSv2, the synapse drivers on v3 are able to modulate it dynamically, which they utilize to implement short term plasticity (STP) (see section 2.1.3). The state variables for the STP dynamics are kept on an array of capacitors within the synapse drivers of HICANN-DLSv3. The corresponding circuits were designed and implemented in the scope of Sebastian Billaudelle's Master's thesis (Billaudelle, 2017), where extensive further details can be found.

Synaptic correlation measurement

In addition to the mentioned mechanisms related to synaptic action (i.e. the address and weight memories and the current DAC), the synapse circuits also contain sensors for measuring spike-timing dependent correlations (Friedmann et al., 2016). In particular, each synapse contains two measurement units, one for the correlation between the pre- and postsynaptic, and another one for the correlations between the post- and presynaptic spike times. Both measurements are represented as analog voltages, V_{cc} and V_{ca}, each stored on a dedicated capacitor. In the first case, the voltage is proportional to

$$V_{\mathrm{cc}} \propto \left(1 - e^{\frac{-(t_{\mathrm{post}}-t_{\mathrm{pre}})}{\tau_{\mathrm{cc}}}}\right), \tag{3.1}$$

where τ_{cc} is a configurable time constant and t_{post} and t_{pre} denote the time of the post- and presynaptic spike, respectively. The index of this voltage (i.e., cc) denotes the fact that it measures the **c**ausal **c**orrelation between the presynaptic cause and the postsynaptic reaction.

V_{ca} on the other hand reflects the **a**causal relationship between a postsynaptic spike and next presynaptic event as

$$V_{\mathrm{ca}} \propto \left(1 - e^{\frac{-(t_{\mathrm{pre}}-t_{\mathrm{post}})}{\tau_{\mathrm{ca}}}}\right). \tag{3.2}$$

t_{pre} is of course directly available to each synapse and t_{post} is provided by a dedicated line that carries the binary spike signal from the neuron circuit up through its entire associated column.

3.1.2 Neurons

The 32 neurons emulate the LIF model (see section 2.1.2) with a current-based synaptic input. A block-level schematic of the entire neuron circuit as presented by Aamir et al. (2016) is given in figure 3.3. The data flow in this figure tends to proceed from left to right.

Synaptic input

First, synaptic input is collected from the two lines that stretch over an entire synaptic column, i.e. 32 synapses, from top to bottom. The injected synaptic current pulses result in a voltage signal $V_{\mathrm{syn,ex/in}}(t)$ similar to the one in figure 2.10, only that the pulses all share the same polarity. Since the current pulses last only for a few nanoseconds, the rise times of the voltage responses are negligible. The decay time constants on the other hand are on the order of a few microseconds and defined by resistors, one for each line, which can be configured by the allocated bias currents `i_bias_syn_res_ex` and `i_bias_syn_res_in`. These voltage signals, on both lines, are then translated into proportional currents by the operational transconductance amplifiers (OTAs) of the synaptic inputs. Signals on the excitatory line result in positive currents, signals on the inhibitory line in negative currents onto the membrane capacitance C_{mem}. Both synaptic inputs can be disconnected from C_{mem} by two switches for debugging purposes.

Leak term and LIF behavior

Another OTA connected to C_{mem} emulates a tunable resistor to the leak potential V_{leak}. Together with C_{mem} it defines the membrane time constant τ_{mem}. C_{mem} is

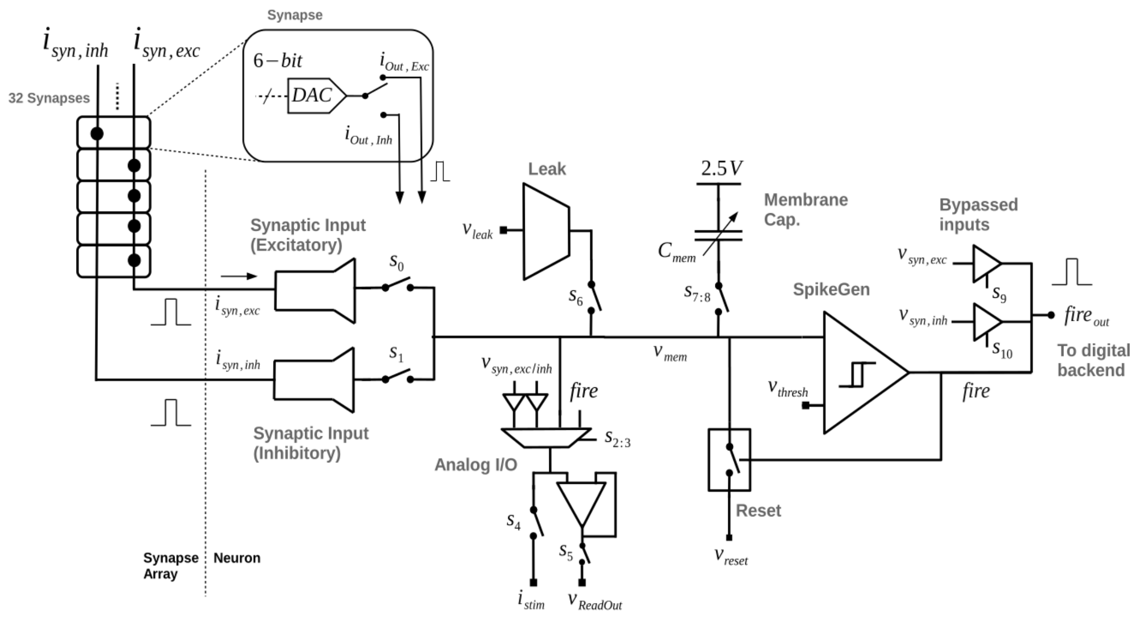


Figure 3.3: Block schematic of the HICANN-DLS neuron. Taken from Aamir et al. (2016).

also digitally configurable but set to a constant maximum value of $C_{\text{mem}} \sim 2.3\,\text{pF}$ for all experiments conducted within this thesis. The circuits listed so far (synaptic inputs, C_{mem}, and leak OTA) together emulate the continuous part of the LIF equations described in equation (2.1). The discontinuous part that is responsible for the membrane reset and hence the action potential (equations (2.3) and (2.4)) is implemented by the spike generating comparator and a corresponding switch that can short C_{mem} to V_{reset}. If the voltage on C_{mem}, i.e. V_{mem}, crosses V_{thresh}, this comparator closes the switch for a certain amount of time τ_{ref} defined by the refractory bias current `i_refr`. Moreover, a spike event is then transmitted to the digital back end.

In HICANN-DLSv3 two additional circuits were connected to C_{mem} that emulate the adaptive and exponential terms of the AdEx model (see section 2.1.2). However, since this functionality is not involved in the experiments discussed in this thesis, it is not described in detail here. For more extensive descriptions the reader is kindly referred to Aamir et al. (2017).

Readout

V_{mem}, as well as $V_{\text{syn,ex}}$ and $V_{\text{syn,in}}$, can be applied to a high impedance amplifier in the analog I/O circuit for external monitoring. This circuit also offers direct access to C_{mem}, such that also voltage or current stimuli can be applied from outside the chip.

3.1.3 Correlation ADC

The synaptic correlation voltages can be read out by a parallel 8 bit ADC that is located at the top edge of the synapse array and stretches horizontally over all columns. As one of its main purposes is the digitization of the voltages accumulated by the synaptic correlation sensors, it is referred to as correlation analog to digital converter (CADC). Every CADC column contains two ADC channels (for the causal and acausal signal) that can convert simultaneously. Hence, 64 input signals can be digitized in parallel. The CADC in HICANN-DLSv2 has been developed within an HBP collaboration between Sabancı Üniversitesi in Istanbul and the University of Heidelberg. Initially, this implementation had some defects, which collectively could have rendered the entire CADC dysfunctional. However, most of these flaws could be mitigated by the Heidelberg group before tape-out. To save complexity in the individual channels, the ADC is designed in a ramp-compare architecture with a centralized ramp generator. This means that a global circuit produces a linearly rising voltage ramp that is distributed to all channels where it gets compared to the input voltages. The time between the start of the voltage ramp and the moment, when the input voltage matches it, is therefore proportional to the voltage difference between the two. This time is measured by counters that are also represented per channel. The maximum sampling rate is therefore naturally derived from the digital clock speed, i.e. 100 MHz, divided by the maximum number of counts, i.e. $2^8 = 256$, an thus, approximately 390 kHz on this chip version. The digital output of the CADC is accessible by the vector unit of the PPU in a way similar to the synaptic weights and addresses.

Due to the aforementioned flaws and weak analog characteristics in v2, the CADC was largely redesigned for HICANN-DLSv3 and HICANN-X. Apart from layout and circuit optimizations in its analog part, the most important change was that all channels were provided with individual digital calibration circuits to mitigate inter-channel variation. Since these changes were all conceptualized and applied by the author, a dedicated and more elaborate description of the CADC follows in section 3.5.

3.1.4 Analog parameter storage

The control voltages and currents that most of the analog circuits require for biasing, configuration, and reference are generated by the on-chip analog capacitive memory (Hock, 2014). This memory is, like the neurons and synapses, organized in columns, with one additional column for global currents and voltages to the left of the neuron-aligned array. Both currents and voltages are stored at a 10 bit resolution. Like almost all other components on the chip, the capacitive memory can be configured by the PPU or by external digital access.

Except for the digital neuron back end, the following components are not part of the ANNCORE and all of them were newly introduced in HICANN-DLSv3. The bonded v3 die is depicted in figure 3.4.

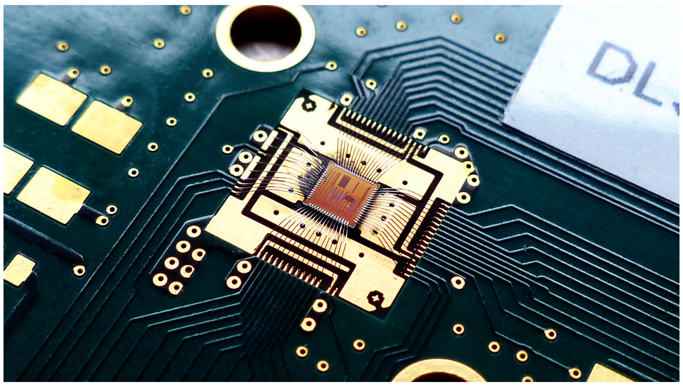


Figure 3.4: HICANN-DLSv3 chip on the chip carrier. The chip measures $1.7 \times 1.7\ \mathrm{mm}^2$.

3.1.5 Membrane ADC

In order to digitize arbitrary ANNCORE signals at high speed and precision a fast successive approximation register (SAR) ADC has been added to the chip. The main motivation behind this was to make the system more autonomous by avoiding the dependency on off-chip measurement equipment. The name membrane analog to digital converter (MADC) derives from one of its major tasks, i.e., digitizing the membrane potentials. It achieves a sample rate of up to 62.5 MS/s, when driven by a 750 MHz clock signal at a resolution of 10 bit. The digital output values are transported similarly to spike signals such they can seamlessly be recorded in a dedicated spike memory. This ADC is the result of a fruitful collaboration with a HBP partners from the EPFL in Lausanne.

3.1.6 Analog readout

Since the MADC has a differential low-impedance input stage, a custom single-ended-to-differential converter with a comprehensive multiplexing and switching input stage first collects and buffers the signals from the ANNCORE. This complex of components is referred to as the readout chain. Comprehensive details on it can be found in Kiene (2017).

3.1.7 Digital neuron back end

Within his Master's thesis, Kiene (2017) also added a digital back end to the analog neuron circuits, which greatly extends their capabilities. Moreover, some functions that were formerly implemented as analog circuits were replaced by more robust and versatile digital units. For example, the neuron refractory period in HICANN-DLSv3 is determined in a counter-driven, digital manner, allowing for more precision, flexibility, and range. To maintain analog signal integrity in the analog neuron circuits and the upper part of the capacitive memory, the digital neuron parts are at the bottom of the capacitive memory and connected only over a few relevant wires. For an exhaustive description of the digital neuron back end, the reader is again kindly referred to Kiene (2017).

3.1.8 Phase-locked loop for internal clock generation

To generate high-speed clock signals for the MADC, the PPU, and the off-chip communication infrastructure, a phase-locked loop (PLL) has been added. A PLL is a control module that converts a reference input frequency to different, mostly higher, output frequencies. The PLL on HICANN-DLSv3 is configurable via JTAG and relies on an external 50 MHz input signal, from which it derives various clock signals with up to 1.5 GHz. Since an initial configuration is required to start the PLL and since core components of the chip depend on its function, HICANN-DLSv3 can only be booted via JTAG. The component is implemented fully digitally and was also developed within a HBP collaboration with the Dresden University of Technology.

3.1.9 Communication and FPGA board

Both prototypes, v2 and v3, communicate with an external field-programmable gate array (FPGA) via four digital data signals and one clock line. On both sides, a *SerDes* is used to (de)serialize the exchanged data. The FPGA sits on the Flyspi board, a custom printed circuit board (PCB), designed by Johannes Schemmel. It contains a Xilinx Spartan-6 FPGA, 2×128 MB DDR3 random access memory, a differential input ADC that can capture 125 MS/s at 12 bit resolution, multiple digital general purpose I/O lines, and USB conncectivity (Hartel, 2016; SP9 partners et al., 2020). This board generally interfaces the HICANN-DLS with external hardware like the host computer, but it can also control the experiment flow and record spike events that are produced during an experiment for later readout.

3.1.10 Summary

Figure 3.5 provides a schematic overview of all the components mentioned so far. Blocks highlighted in yellow were introduced with v3, while the others were already present on v2. Most of the components that were added on the transition from HICANN-DLSv2 to v3, could be tested successfully after tape-out and turned out to be widely functional. Some minor flaws and errors became well understood during testing and could be further improved for the first upscaled version. However, a problem in the digital communication system caused the chip to crash in the case of high spike traffic, rendering the chip unusable for larger, especially recurrent network experiments. The exact origin of this bug remained unresolved but was suspected to be related to clock transitions induced by the newly introduced varying PLL clock domains. As most of the communication infrastructure was replaced in the transition to the full-sized chip, however, no further effort was taken into clarifying the exact causes. Therefore, network experiments, among them the experiments in chapters 4 and 5, were continued on HICANN-DLSv2 instead, until the new setup incorporating the full-sized HICANN-X became available.

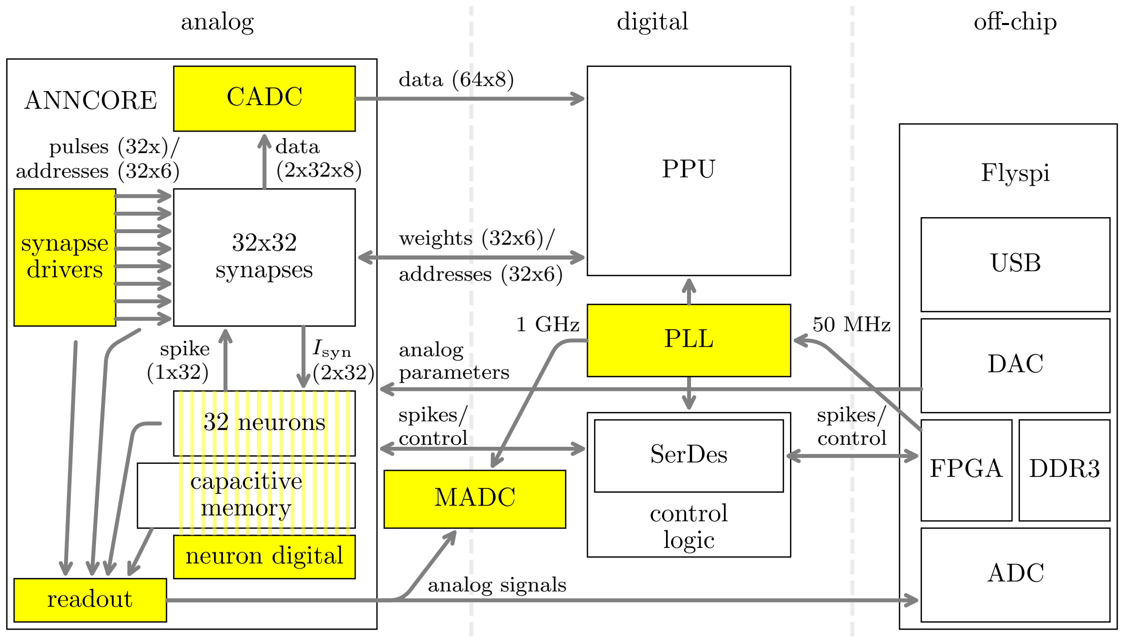


Figure 3.5: Logical block-level diagram of the HICANN-DLSv3 setup. *Analog* and *digital* are both part of the chip, while the right *off-chip* section contains components that are implemented externally. Highlighted in yellow are the components that were not yet part of HICANN-DLSv2 and new to v3.

3.2 Prototype test setup

The connection between the prototype chips and the Flyspi is realized by a set of two PCBs, the *baseboard* and the *chip carrier board*. The latter provides a landing site for the HICANN-DLS chip and has the form factor of a Small-Outline Dual Inline Memory Module (SODIMM). It is mostly equipped with passive components for supply voltage stability and some test pads for direct signal readout. Its current implementation is based on a design by Matthias Hock and was slightly modified to satisfy the HICANN-DLSv3 requirements. It can be inserted into a standard SODIMM connector on the baseboard.

The baseboard on the other hand required a more significant modifications on its transition from v2 to v3 but was designed to be fully backward compatible with v2. The redesign was mostly necessitated by the newly added JTAG interface as well as the change in the chip's clocking scheme. Due to several other introduced improvements, the v3 baseboard eventually superseded most of the older v2 baseboards after its introduction. The functional components and the relations between them are depicted in figure 3.6 and described in detail below.

3.2.1 Power supply

The HICANN-DLS requires two different supply voltage levels, i.e. 1.2 V and 2.5 V, each for both, the digital and the analog domain. This domain separation is introduced to shield the analog supply voltages from noise and distortion induced

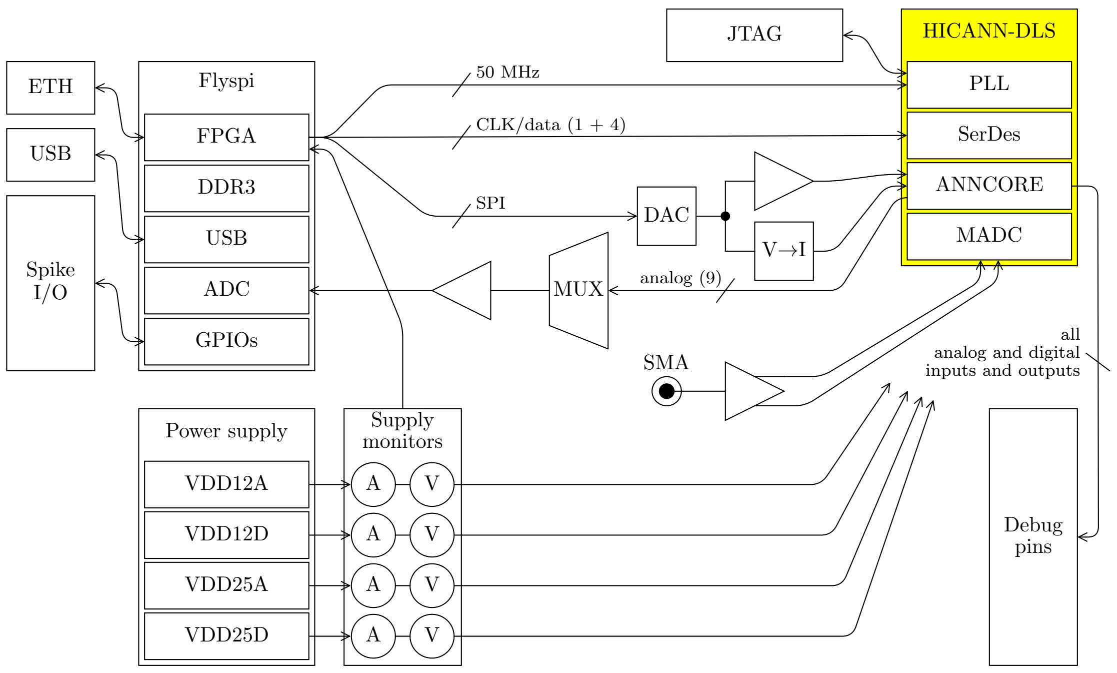


Figure 3.6: Block-level schematic of the baseboard for the HICANN-DLSv3. This setup is also compatible with the 2nd generation prototype.

by the fast switching digital circuits. The latter operate almost exclusively on 1.2 V, while the digital I/O cells are driven with 2.5 V mostly. In the analog domain, 2.5 V are usually avoided for power efficiency reasons but can occasionally be of advantage. In particular in scenarios where voltage operations close to the 1.2 V supply rail are desired. In these cases, introducing 2.5 V can often help avoiding signal distortions or circuit complexity that have to be tolerated otherwise.

The four voltages, `VDD12D`, `VDD12A`, `VDD25D`, and `VDD25A` and the corresponding supply currents can be monitored by a bank of four `INA219` current sense amplifiers (Tex, 2015b) from the Flyspi FPGA.

3.2.2 MADC test input

Since both, the readout chain and the MADC were newly introduced on this prototype chip, a versatile test interface for both became part of the baseboard. Essentially, a shielded test signal can be connected to a single-ended to differential amplifier that can be switched directly to the MADC inputs. The same signal chain also contains a few test pads and removable 0 Ω resistor bridges such that analog data from the ASIC can also leave the chip and be accessed from the outside. Kiene (2017) contains some characterizations and measurements that rely on this unit.

3.2.3 ADC chain

Single-ended analog voltages can also be digitized by the Flyspi ADC. For that purpose, the signals are first selectably routed through a 9-channel multiplexer composed of three smaller low on-resistance 3:1 `TS5A3357` multiplexers (Tex, 2018a) before they are pre-buffered by a `LMH6612` rail-to-rail operational amplifier (Tex, 2013). This preamplifier shifts and scales the signals to the appropriate input voltage of a `LMH6518` programmable gain differential output amplifier (Tex, 2016) that is fast and low-impedant enough to drive the Flyspi ADC. Both the multiplexer and the differential amplifier are configurable by the FPGA.

3.2.4 Reference signals

Being in the developmental stage, many circuit components on the prototype chips have to receive reference voltages and currents from outside the chip. These signals are generated by an `AD5328` DAC (Ana, 2016) and some operational amplifiers and custom voltage to current converters.

3.2.5 Spike I/O

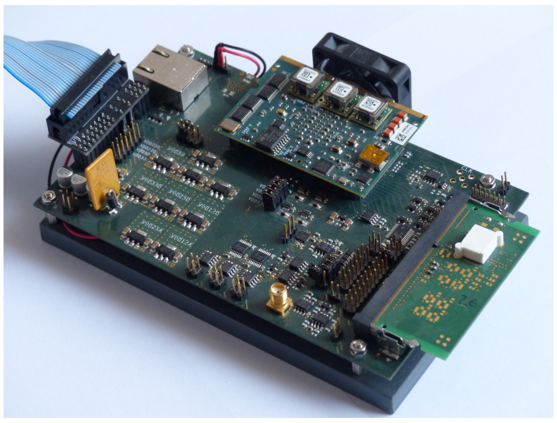
A few general purpose pins of the FPGA on the Flyspi board are accessible via an array of `TXB0108` bidirectional level shifters (Tex, 2018c). These integrated circuits (ICs) do not only shift the FPGA voltage levels into more commonly used ranges, simplifying external access, but also protect the fragile FPGA pins from electrostatic discharge and other electrically related hazards. On the baseboard, these signals are guided to an array of pin headers for easy access.

These pins play a key role for the robotic experiments which are part of later chapter 4, where they are used for spike input and output to and from the analog core of HICANN-DLS. The FPGA design was extended to assign the first six neurons in the neuron array one pin on the mentioned pin header where every spike results in a digital pulse with a width of 500 ns.

Signals to the HICANN-DLS are encoded in a similar way. When a rising edge appears on one of the remaining seven pins on the header, the FPGA injects a spike event into the corresponding one of the top seven synapse rows in the synapse array. The address of these spike events can be freely configured. The necessary FPGA hardware description was written and implemented by Christian Pehle.

3.2.6 Spike router

This Spike I/O mechanism is technically coupled to the spike router, which also resides in the FPGA. This unit, which was also implemented by Christian Pehle, is responsible for recurrent routing. If it is activated, spikes leaving the HICANN-DLS are assigned a configurable target address and row and sent back into the chip.



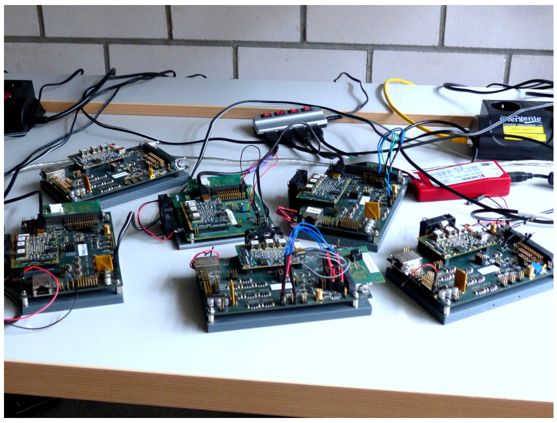


Figure 3.7: Left: Photo of the HICANN-DLSv3 baseboard with the HICANN-DLSv2 chip carrier board (plugged in at the right). The Flyspi FPGA board sits on top of the baseboard (middle, close to the fan) and a ribbon band cable is connected at the left via a small adapter PCB. This setup has been used for the experiments described in later chapters 4 and 5 of this thesis. Right: A flock of v3-baseboards and one old v2-baseboard among them. The setups retired when experiment development for HICANN-X started.

Lastly, the baseboard contains an Ethernet interface that has, however, never been commissioned. Generally, all signals on the board, digital as well as analog are also accessible by test pads or pins for additional debugging support. The base board is fully compatible with the HICANN-DLSv2 and its chip carrier board. All later described experiments have, unless explicitly noted, been carried out on this setup. The entire schematic can be found in appendix C.1 and the setup is depicted in figure 3.7.

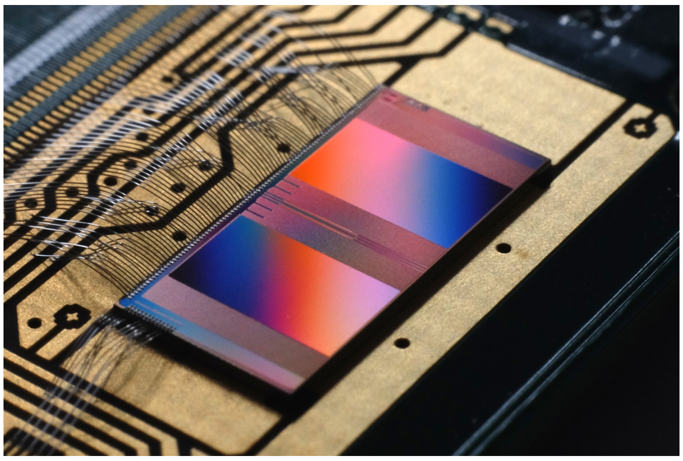


Figure 3.8: HICANN-X chip on its chip carrier. The chip measures $8 \times 4\ \mathrm{mm}^2$.

3.3 HICANN-X

The HICANN-X is the first full-scale version of the BrainScaleS-2 series with 512 neurons with 256 synapses each, i.e., 131072 synapses in total. Besides stating a significant milestone in the BrainScaleS family, it is the first autonomous analog neuromorphic chip with digital multicore plasticity management. While many of the HICANN-DLSv3 components were simply scaled up to the larger array size, some received additional updates. Moreover, the digital infrastructure changed significantly. Below listed are the most important changes as compared to the previous incarnations.

3.3.1 ANNCORE and quadrants

Figure 3.9 gives an overview of the arrangement of the ANNCORE (blue) and the digital components surrounding it. Apart form the synapse drivers and the digital neuron circuits, the ANNCORE is composed of four quadrants that each contain an instance of the CADC, the synapse array, the analog neuron part, and the capacitive memory. Each quadrant contains 128 neurons, 256 synapse rows, and hence 32768 synapses in total. Like in the prototype chip, there are two CADC channels per neural column. Figure 3.10 depicts an elaborate block schematic of one quadrant with all relevant digital and analog components and data buses.

Synapse drivers and the machine learning mode

The synapse drivers blocks sit in the middle between two horizontally aligned quadrants and are responsible for both, the left and right synapse arrays. Each of the 128 drivers within each block is individually addressable and responsible for two rows, i.e., 2 sides $\times$ 2 rows $\times$ 128 columns $=$ 512 synapses. Since the 6 bit synaptic address space allows only for 64 different target addresses, a spike has to be received by at least eight neurons simultaneously if all synapses shall be used. This limits the maximum number of different presynaptic partners to 16384 per chip. In HICANN-X, these drivers were extended by a machine learning mode in which the STP pulse length modulation is reused to encode an arbitrary value. I.e., five of the

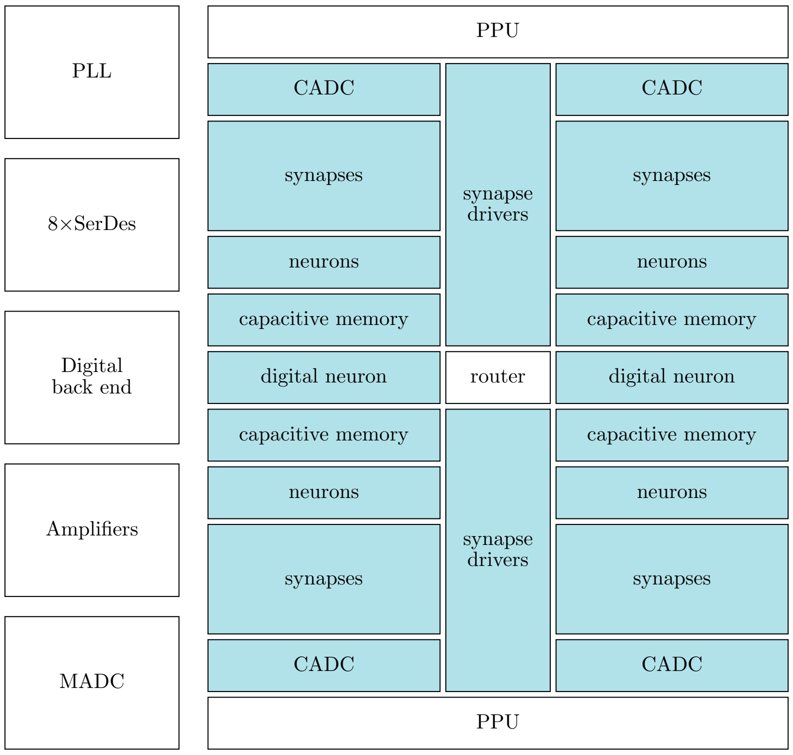


Figure 3.9: Components of the HICANN-X and arrangement of PPUs and ANNCORE. The elements that are part of the ANNCORE are blue.

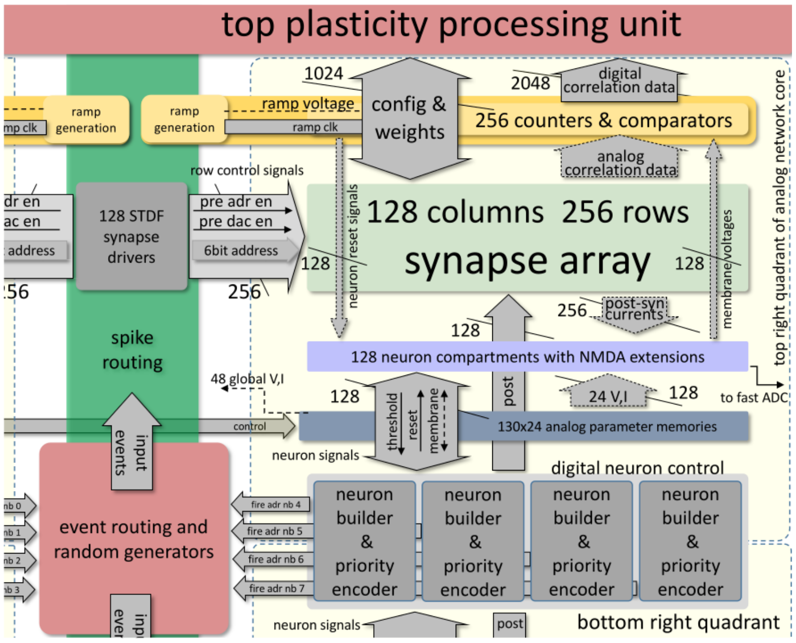


Figure 3.10: Block schematic and data flow of the top right quadrant of the HICANN-X ANNCORE. Taken from Schemmel et al. (2020).

six bits of a spike package that usually encode the sender address, encode the pulse length instead. In this way, the synapses inject a charge into the synaptic input line that is proportional to the product of the weight and this received five bit value, and hence, they implement an analog multiplication. As neurons can be brought into a state where they integrate this injected charge over time, the entire synapse array is transformed into a vector-matrix multiplier.

Event routing and random generators

The digital neuron circuits are located in the middle between each two vertically aligned quadrants. The circuits belonging to the upper quadrant are interleaved with those from the lower quadrant, forming one block. A bus system at the center of it, transports the spike events to the middle of the ANNCORE, where a synthesized event router is located. This unit merges and splits input to output signals and streams spike events to the top and bottom into the synapse drivers. The router block also contains spike sources that can directly inject spike trains with random Poisson-distributed or equidistant inter-spike intervals into the neural network. Since Poisson noise is commonly used to provide random background activity for neural networks, and since this type of noise intrinsically has a high data bandwidth, these generators save a considerable amount of off-chip communication resources.

3.3.2 Other components

The ANNCORE is surrounded by two PPUs, one at the top and one at the bottom, such that both have high-speed parallel access to the corresponding CADCs and the synapse arrays. To the left of the ANNCORE are the PLL, the MADC, two analog output buffers (amplifiers), eight high-speed SerDes links, and most of the digital back end. Both the PLL and the MADC were taken over without updates from the HICANN-DLSv2/3 prototypes. The amplifiers could also be reused from previous chips and sit, like the MADC, behind a large multiplexing stage that can route analog signals from and to the analog core. Moreover, HICANN-X contains an internal reference current generator from which all internal biases and reference voltages are derived. Thus, after initial calibration, the chip is completely independent of external analog signals.

Another major change concerns the data interface (Karasenko, 2020). Instead of previously 4 single-ended data pins, the upscaled version contains eight bidirectional full-duplex differential high-speed links, each with a data rate of 1 Gb/s output plus 1 Gb/s input, when driven with a 500 MHz clock signal. The total I/O bandwidth is therefore up to 8+8 Gb/s. With the current spike encoding scheme, this bandwidth results in a maximum rate of 250×10^6 spikes per second for each direction. To put this into intuitive relation, the maximum allowed output spike rate per neuron is then about 488 kHz. This is roughly five times above what is maximally expected (100 kHz per neuron) and about 50 times above the average operation (10 kHz per neuron). The output bandwidth is therefore amply sufficient.

Concerning the input bandwidth, a maximum of 16384 presynaptic partners (constrained by the address space per row and the number of synapse drivers) could fire at a maximum rate of 100 kHz. This extreme case would lead to roughly 1.6×10^9 spikes per second and therefore exceed the technical capacity by a factor of about 6.5. In the average case, however, the input rate is 10 times lower, i.e., about 164×10^6 spikes per second, and therefore well within the link capacity. Also note that, the larger the network is, the less likely it is that all neurons spike at maximum rate. Moreover, the link capacity can be doubled if clocked with 1 GHz instead of 500 MHz. The latter is only chosen because of technical limitations imposed by the FPGA that the chip is currently connected to (Karasenko, 2020). This interface between the chip and the FPGA is physically implemented on a PCB, which was also developed within the context of this thesis.

3.4 Full-scale test setup

The current HICANN-X test platform is based on the *Cube setup* (Güttler, 2017; Kleider, 2017). These setups are usually composed of six PCBs: four Kintex-7 FPGA boards, one *IO-Board* for external I/O, like Ethernet and USB, and one *iBoard-Adapter* for interfacing to the HICANN chip carrier boards. They were originally developed for prototyping the HICANN-to-FPGA connectivity for the BrainScaleS-1 wafer-scale system, which contains the same but more of these Kintex-7 boards. Facing the same challenge again with the BrainScaleS-2 prototype chips, reusing these setups came as a natural choice. Hence, a new board was developed to serve as a bridge between the Cube setups and the HICANN-X chips. Since the BrainScaleS-1 equivalent of this PCB is the *iBoard*, the HICANN-X version is called *xBoard*.

Figure 3.11 shows a Cube setup with the new PCB installed. Mechanically, it is held in place by some custom mounts that attach to the vertical aluminum rail at its back. The xBoard is connected to the iBoard-Adapter by a 120-pin Samtec `BTH-60` connector that is mostly occupied by the 18 differential pairs for the high-speed link and a large number of ground connections for signal integrity. Moreover, it carries a few further differential signals for a high-speed ADC that sits on the xBoard, and some other signals for slower communication. Details and components are listed below and the entire schematic can be found in appendix C.2. Figure 3.12 gives a block-level schematic overview of the xBoard.

3.4.1 Power supply

The arrangements of linear regulators and `INA219` current and voltage monitors was largely adopted from the HICANN-DLSv3 baseboard and extended by two further supplies, for the MADC and the PLL. In this way, these components can be separately monitored for their power consumption while the system remains fully functional. Moreover, all of the chip supply voltages can be fine tuned by a DAC that can take influence on the corresponding linear regulators. Therewith, the HICANN-X can be tested for its capability of operating at lower supply voltages and thus, at greater energy efficiency. A software-defined feedback loop that sets the voltages based on the `INA219` data was implemented by Mitja Kleider.

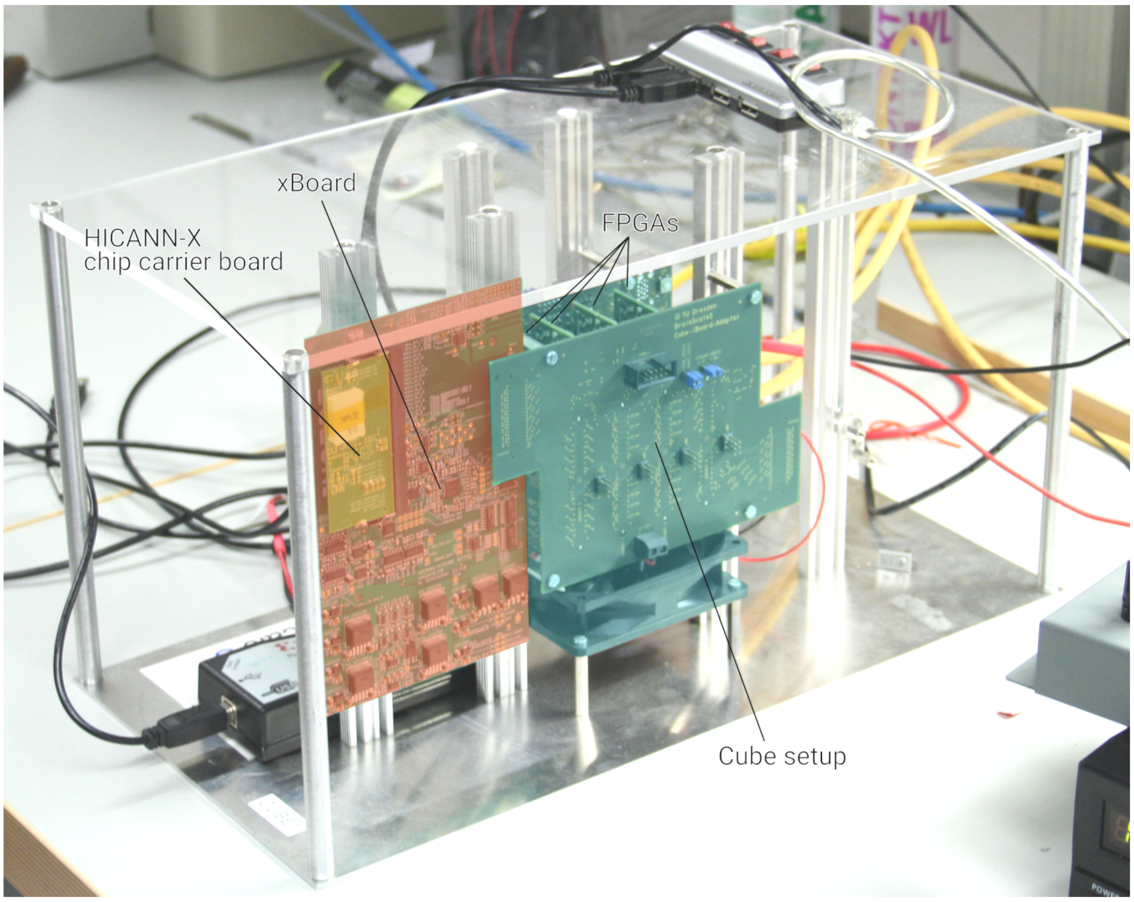


Figure 3.11: Cube setup with xBoard and HICANN-X chip carrier. The components of the legacy Cube setups are have a green overlay, and the new HICANN-X components are tinted red and yellow. The IO-Board connects the four FPGA boards and is only partially visible in the background. The iBoard-Adapter is the z-shaped PCB in the foreground.

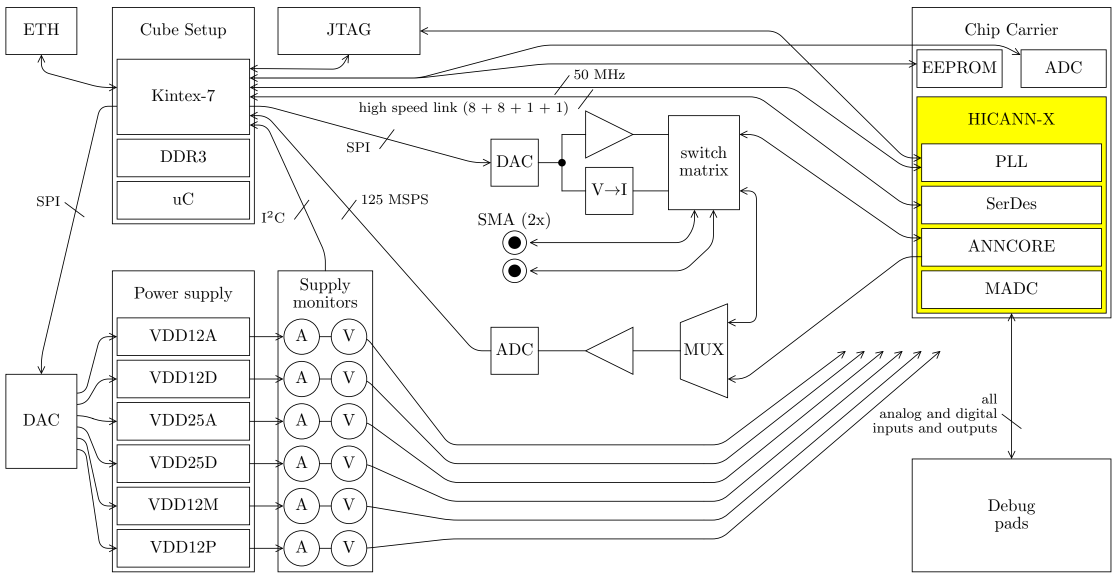


Figure 3.12: Block-level schematic of the xBoard. The HICANN-X chip is highlighted in yellow.

3.4.2 Readout chain and reference generation

Also some of the analog readout multiplexing circuits from the HICANN-DLSv3 baseboard could be reused for the xBoard but were extended to a certain degree, mostly because the analog HICANN-X pins can be configured both as inputs and outputs, depending on the test case. For example, the pin `IREF` can be connected to the newly introduced chip-internal reference current generators for readout and characterization, but can also receive an external reference current as a fallback solution in case of the circuit's failure. The pins `ANAREADOUT_DBG_0/1` can also be configured as inputs and outputs. Therefore, these pins can be gated to readout circuits as well as to buffered DAC outputs. Additionally, some signals can be accessed by two SMA connectors for shielded external measurements. The readout signal is also passed to a `LMH6518` differential programmable gain amplifier but then digitized locally by a `ADC3224` 12 bit, 125 MS/s high-speed ADC (Tex, 2019), which also streams to the Kintex-7 FPGA.

3.4.3 Digital I/O

The xBoard also contains some level shifters and digital output extensions from which various digital signals for IC control or light-emitting diode (LED) indicators emerge. While a Cube setup contains up to four FPGA boards, only one is used per HICANN-X. The corresponding digital high-speed busses are routed from the Samtec connector, across the xBoard, to a SODIMM connector that receives the chip carrier board. On both boards the wires are matched in total as well as differential length to equalize signal runtimes.

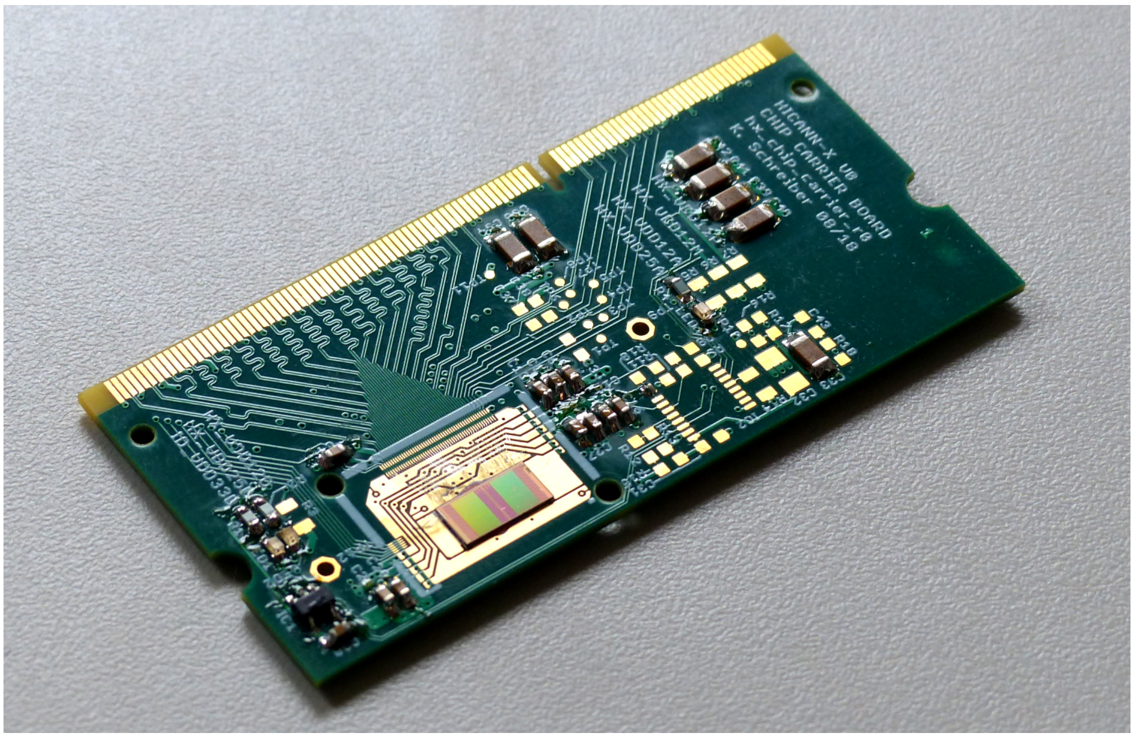


Figure 3.13: Partially assembled HICANN-X chip carrier board.

3.4.4 Chip carrier

Besides the landing site for HICANN-X and some capacitors for supply voltage filtering, the chip carrier (figure 3.13) also contains LED-based supply voltage indicators and two ICs, a `24AA02UID` electrically erasable programmable read-only memory (EEPROM) (Mic, 2013) and a `MAX11615` 8-channel ADC (max, 2012). The `24AA02UID` also contains a unique 32-bit identification number, which can be used to identify the individual HICANN-X chips. Data from central calibration data banks can therefore be accessed automatically and chip-specific data can always be associated with this specific ID. Moreover, 2 kB of EEPROM can be used for arbitrary local metadata on the respective chip. The ADC was originally intended to provide measurements for a supply voltage control loop but has become obsolete by the `INA219` monitors on the xBoard. It therefore remains unassembled.

All peripheral components on both, the xBoard and the chip carrier, are digitally configurable over either a serial peripheral interface (SPI) or an inter-integrated circuit (I^2C) bus that are both driven from the Kintex-7 FPGA in the Cube setup. At the time of writing, 16 HICANN-X Cube setups were actively in use, eight of which are depicted in figure 3.14.

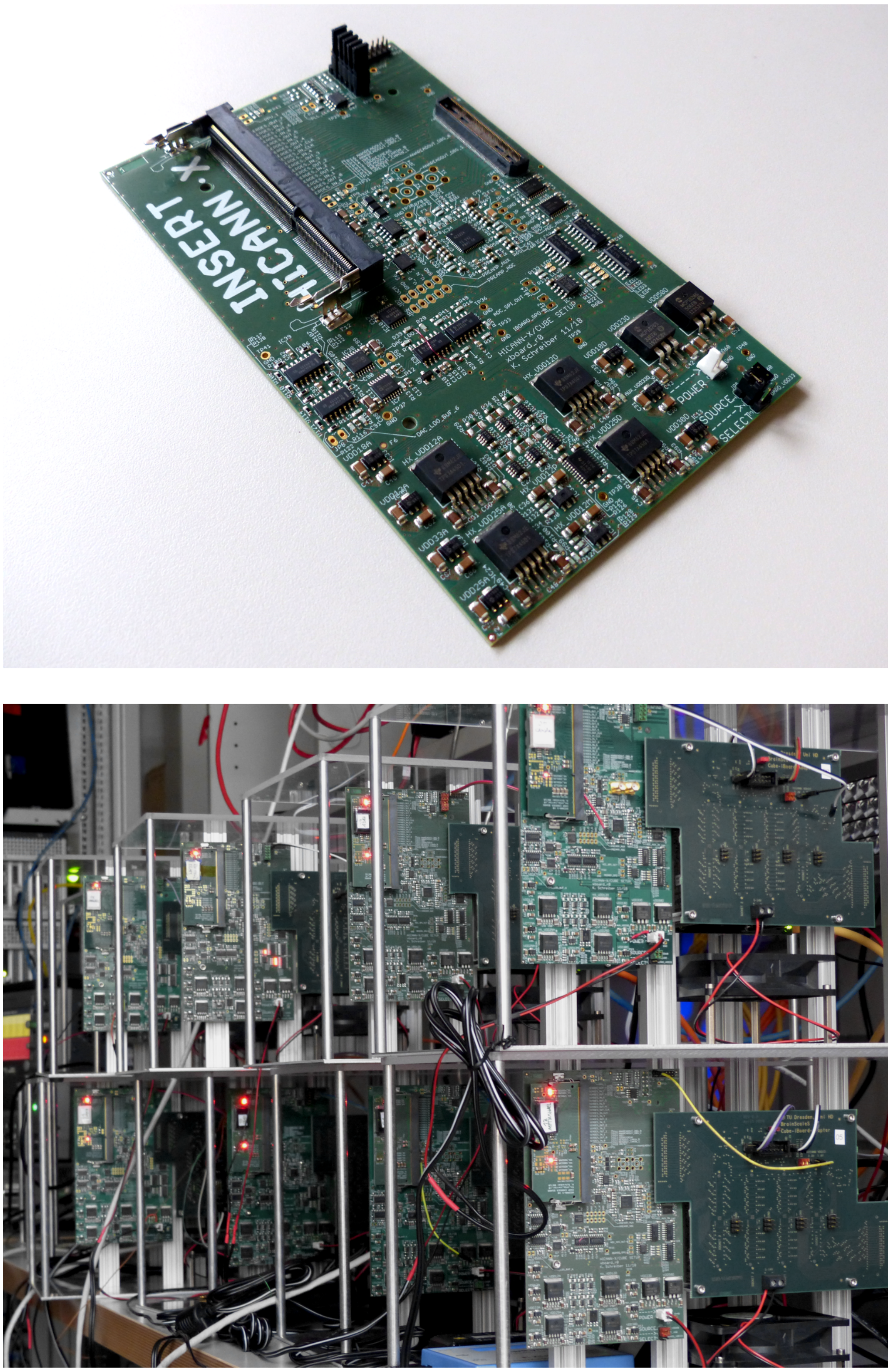


Figure 3.14: Assembled xBoard without chip carrier PCB (top) and a few HICANN-X test setups in fully assembled working state (bottom).

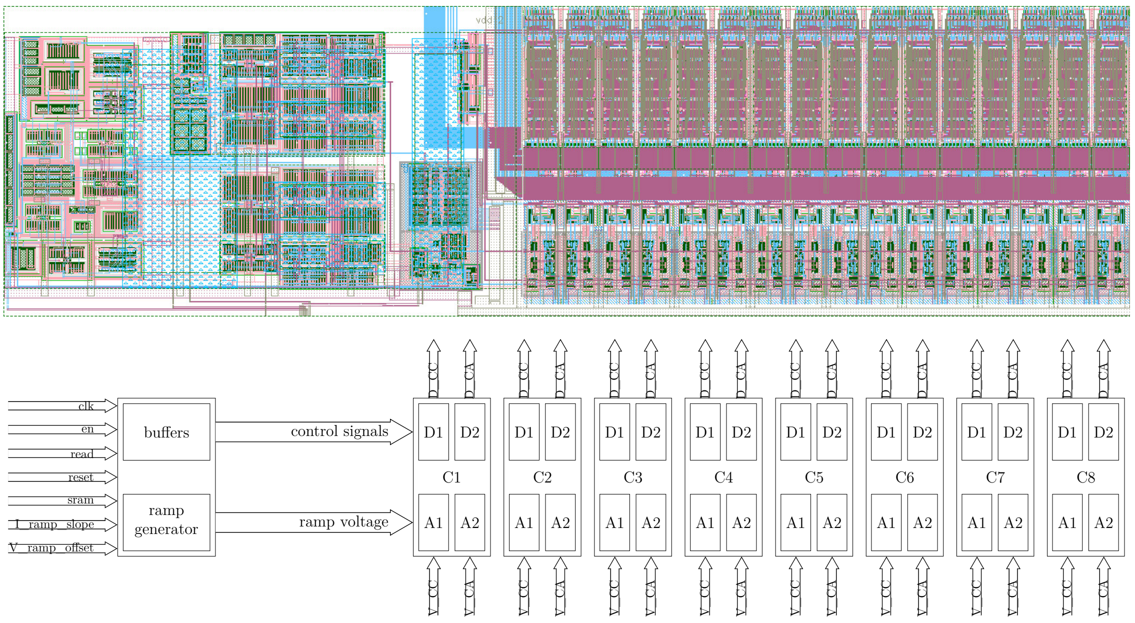


Figure 3.15: CADC architecture. The ramp voltage and a few control signals are distributed from a central section at the left to an array of CADC columns. The central section receives analog references and biases, as well as digital configuration and control signals. Each column is subdivided into two channels for the causal (cc) and anti-causal (ca) STDP correlation measurements, which, in turn, consist of an analog circuit A and a digital circuit D each. The voltage signals to be measured arrive from the bottom of the array and the digitized values leave the CADC at the top into the PPU. The colored top figure shows chip layout of the central section and the first eight columns and the bottom figure a schematic representation of it.

3.5 CADC

The CADC is structured in an array of column-wise organized channels, and a central component that distributes signals which are shared among all channels. This central compound, also referred to as the CADC center, is responsible for the ramp and reset voltage generation and buffered signal distribution. Figure 3.15 illustrates the arrangement of the subcomponents.

3.5.1 CADC center

Parts of the CADC center were recycled from the previous HICANN-DLSv2 design from Sabancı Üniversitesi only with minor modifications. In particular, the operational amplifier, that is responsible for the ramp generation was reused in a different circuit and layout configuration. In both, the new and the former design, the ramp voltage is generated by integrating over an adjustable reference current. For that purpose, the operational amplifier is configured with an integrating capacitor in the feedback path that neutralizes its input current by a constant increase in voltage. In the HICANN-DLSv2 design, this current derives from a resistor network that receives a reference voltage from the capacitive memory. In the current implementation this current is obtained more straight-forwardly by a mirrored reference current from the capacitive memory. In both versions, the ramp reset is executed by shorting the integrating capacitor, turning the entire circuit into a unity gain buffer with the ramp reset voltage at its input.

The CADC center also contains a smaller operational amplifier (taken from the voltage buffers in the capacitive memory (Hock, 2014) that constantly buffers this reset voltage to all CADC channels. The reason for this is that the ramp voltage wire has a large parasitic capacitance attached to it because it travels over a distance of 1.5 mm through an entire quadrant and because it is attached to large gate capacitors of the comparators in every channel. In order to accelerate the reset process, every channel has a switch that can short the ramp to the reset voltage, thus, maximizing the reset current.

In addition to the analog voltages, most of the digital signals necessary to clock and control the CADC channels are also buffered in the central section.

3.5.2 Columns and channels

As mentioned earlier, voltage conversion in a ramp-compare ADC is accomplished by measuring the time it takes for a constantly rising ramp voltage V_{ramp} to exceed the input voltage V_{in}. To measure this time, every channel contains a digital counter that is incremented as long as $V_{\mathrm{ramp}} < V_{\mathrm{in}}$. This is achieved by gating the counter's clock signal with the output of an analog comparator that continuously compares the two voltages. The crucial aspect in this scheme is the input voltage mismatch of the comparator, which must be kept low in order to minimize inter-channel variations. A common way to do so, is to maximize the gate area of the input transistors, because this minimizes the relative impact of unavoidable manufacturing width, density, and thickness variations with respect to the total area of the component.

However, each channel is subject to significant area constraints, as it is instantiated 1024 times on one chip. Therefore, besides area upscaling, a calibration mechanism has been implemented, which compensates 0^{th} order mismatch, i.e., offset variations.

Although the CADC achieves a precision of roughly 8 bit, every counter has a 9 bit range and starts at a configurable value between 129 and 383, usually around 256. If the conversion value is smaller than 256, a readout logic sets the digital output value to zero. Moreover, the most significant bit is truncated, such that the visible results are effectively within 0 to 255, while, internally, the counter runs from around 256 to 511 maximum. Thus, by setting this initial value to the negative offset deviation of the respective channel, the offset mismatch is effectively calibrated.

Appendix C.3 contains the schematics the CADC channel, the comparator, the digital part including the calibration logic, and the ramp generator.

3.5.3 Measurements and performance

Figures 3.17 and 3.18 show the uncalibrated and calibrated conversion response functions. As typical deviations are on the order of ±10 least significant bit (LSB),

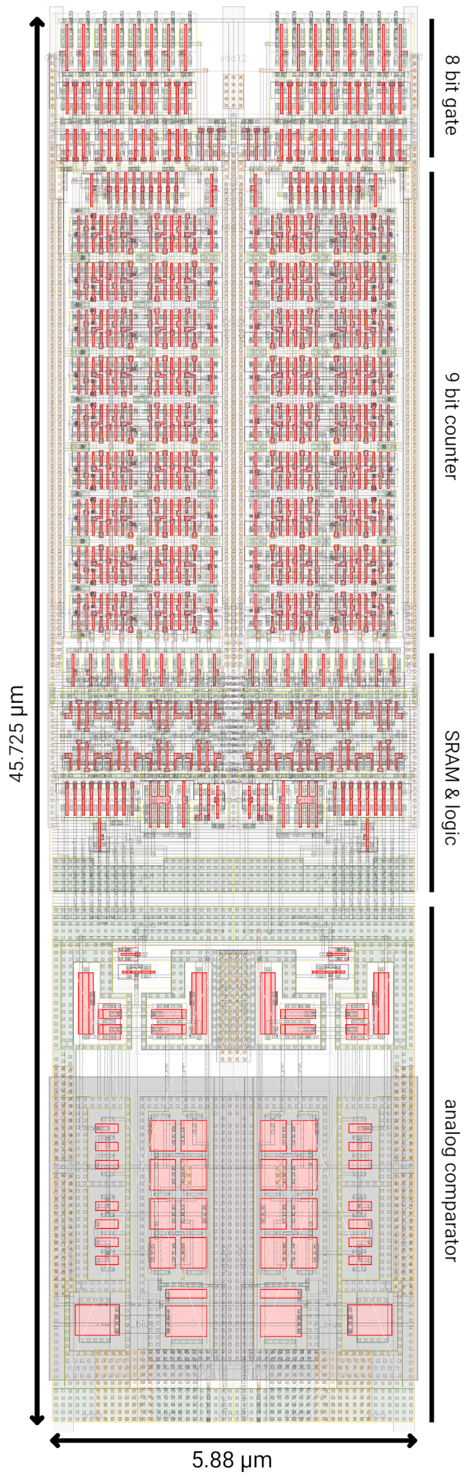


Figure 3.16: CADC column consisting of two mirrored channels. On the bottom are two analog comparators. Two 8 bit SRAMs store the calibration values that define the starting values for the 9 bit digital counters in the upper halve. In order to prevent unnecessary switching currents while counting, the results are gated and only accessed when the conversion is finished. The red areas correspond to the polysilicon of the transistor gates. The digital parts in the upper halve is populated with a high transistor density and small gate sizes, while the analog lower halve contains larger gates for mismatch reduction.

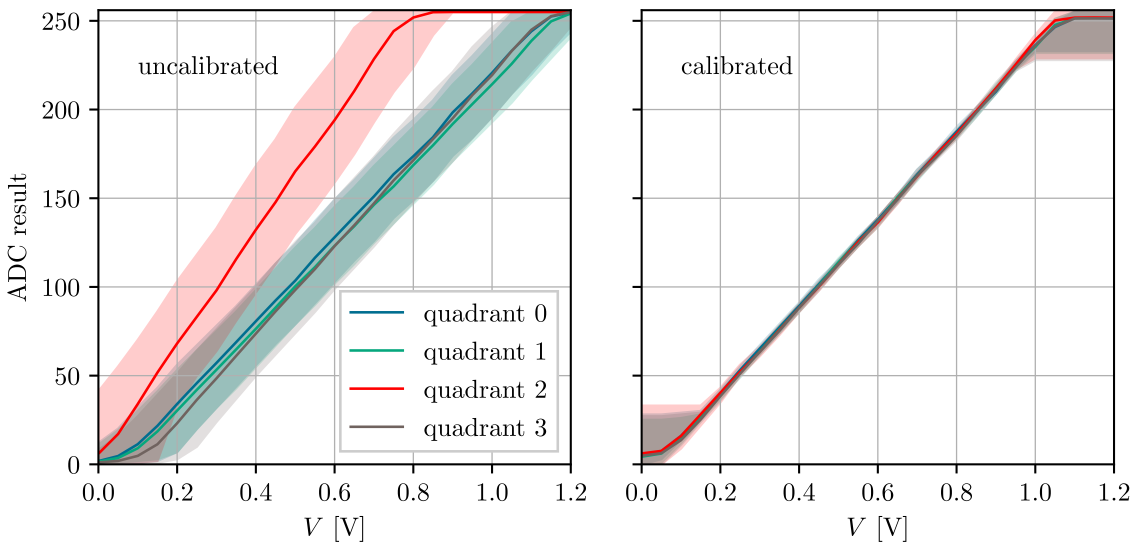


Figure 3.17: Uncalibrated and calibrated CADC response functions for all channels within the four quadrants of one HICANN-X chip. The filled areas are limited by the respective minimum and maximum values. Calibration took place per quadrant for the ramp current that determines the ramp slope, and per channel for the inter-channel variation using the custom digital calibration strategy. The data was generously provided by Johannes Weis, who also implemented the CADC calibration routines in HICANN-X (Weis, 2020).

this calibration scheme causes the range of some channels to shrink by an equal amount because of digital clipping at the lower and upper counter range limits. So far, however, no cases have been encountered in which this limitation stated a problem.

The power consumption imposes another constraint. The initial design goal was a current consumption of 2 µA per channel, when driven from a single 1.2 V supply rail. The corresponding power consumption was therefore targeted at 2.4 µW. Figure 3.19 puts this into relation to in-silico ADCs by plotting the energy per conversion over the ADC resolution for various values found in literature (Sundstrom et al., 2008). The smallest technology included in this publication is a 90 nm process, which uses roughly twice the area for digital circuits than the 65 nm technology used for the presented CADC. For the sake of the argument, a factor two in terms of power efficiency merely due to the technology improvement can be assumed.

The targeted signal-to-noise ratio is somewhere between 6 and 8 bit and the energy consumption for the corresponding supply currents is marked in the figure as a pink (7.3 pJ for the presented solution) and a blue (1.3 pJ, corresponding to 2 µA target current consumption) bar. It becomes apparent that 2 µA per channel is approximately one to two orders of magnitude below the state-of-the-art at the year of the publication, i.e., 2008, while 7.3 pJ is still at the lowest end. Note however, that the included ADCs are not necessarily subject to the tight area constraints that the HICANN-X imposes. Unfortunately, until the current solution percolated, a lot of effort has been wasted for aiming at the desired goal of 2 µA per channel. A

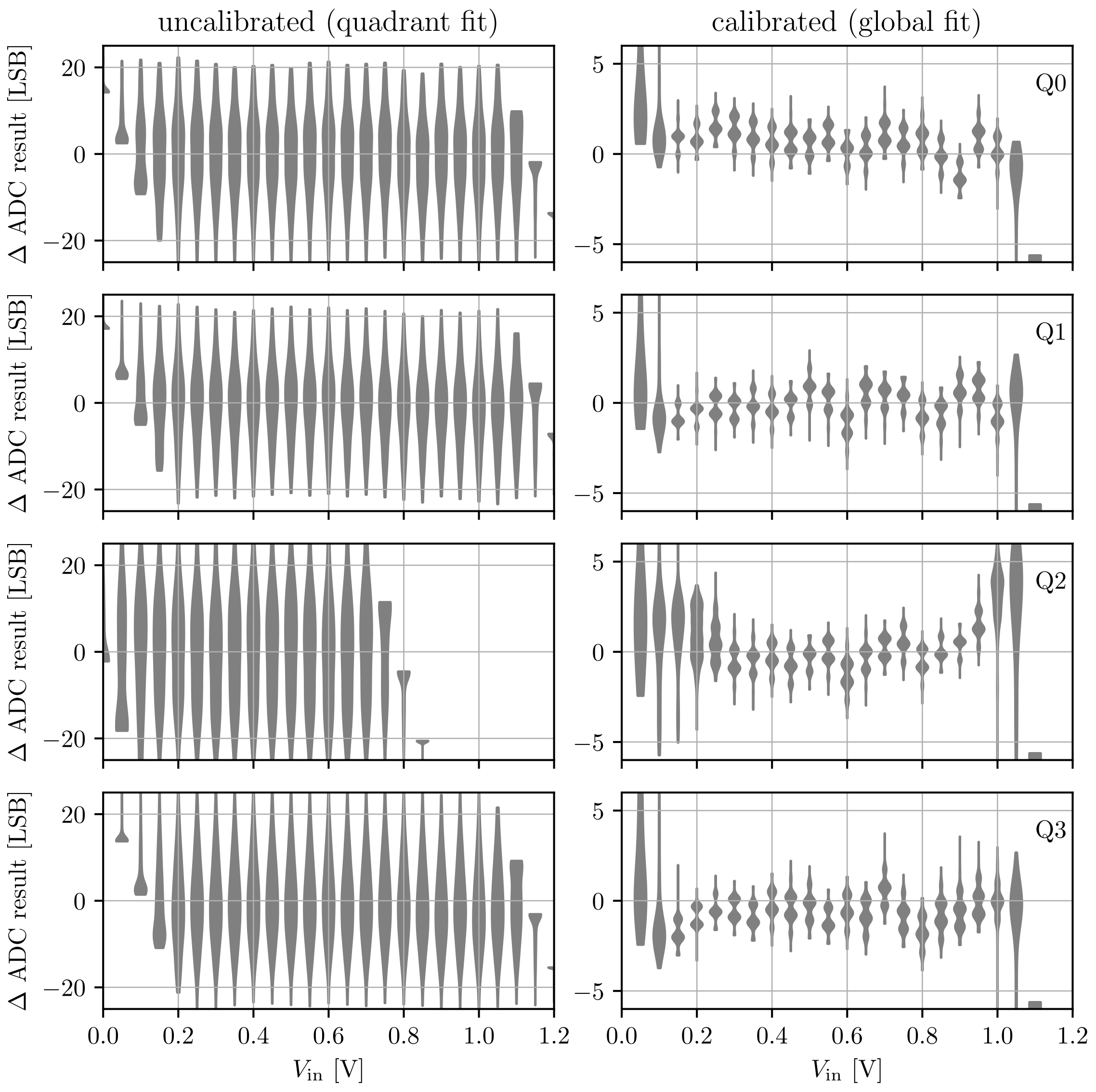


Figure 3.18: Uncalibrated and calibrated CADC deviations for all channels within the four quadrants (top to bottom) of one HICANN-X chip. The violin plots show the histogram over the deviations from a linear fit at the various data points, i.e., the integral nonlinearity (INL). The fits in the left column were obtained by linear regressions based only on the data from the respective quadrant. The fits on the right are obtained from data from all quadrants. These plots are based on the same data set as in figure 3.17.

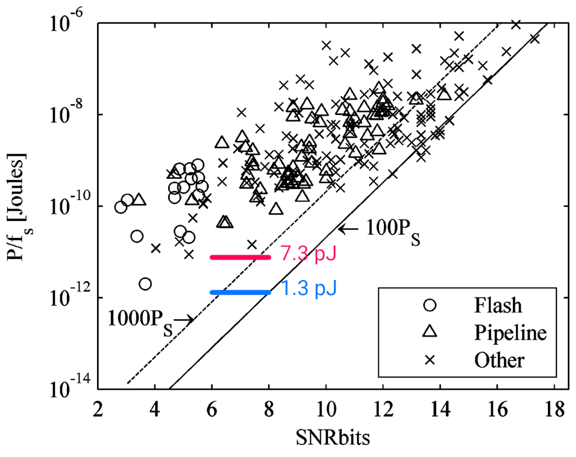


Figure 3.19: Conversion energy with respect to the signal-to-noise ratio for different ADCs compiled from various publications. The blue and pink lines put the CADC conversion energy (1.3 pJ for 2 µA total current consumption and 7.3 pJ for the presented solution) into this perspective. The smallest technology that is reflected in the data points is 90 nm gate length. Taken and adapted from Sundstrom et al. (2008).

design that achieved a total consumption of 2.13 µA per channel in simulation was ultimately dismissed because it demanded periodical calibration, which would have negatively impacted the user experience (Schreiber, 2016).

The current implementation is therefore oriented at a more conservative design and uses 0.5 µA to 1 µA for the analog circuits on 2.5 V and approximately 6 µA for the digital part when running at a maximum sample frequency of approximately 1.85 MHz. Taking into account the power demands of the central stage, another 4.1 µA have to be added per channel on 1.2 V, and thus, the total current consumption per channel is roughly 10.1 µA on 1.2 V and 0.5 µA on 2.5 V. Hence, a conversion takes about 7.3 pJ, which is still well within a state-of-the-art performance margin. A summary of the CADC key characteristics is given below.

Resolution	8 bit
Input range	0 V to 1.2 V (scalable)
Maximum sampling rate	∼1.85 MHz
Nr of channels	256 per unit, 1024 per chip
Power per channel	∼13.4 µW
Energy per conversion and channel	∼7.3 pJ
Power per CADC	∼3.4 mW
Channel height	45.725 µm
Channel width	5.88 µm
Area per channel	268.863 µm^2

The input range is labeled *scalable* because it can be adjusted with the ramp offset and slope. In this way, resolution can be traded against range because the counting time stays constant. For example, if the signal of interest does not go below 0.4 V and not above 0.8 V the ramp can be configured such that it starts at the 0.4 V and reaches 0.8 V, when the counter reaches at 256. This maps the 8 bit output range to 0.4 V to 0.8 V and improves the LSB resolution from 4.7 mV to 1.6 mV.

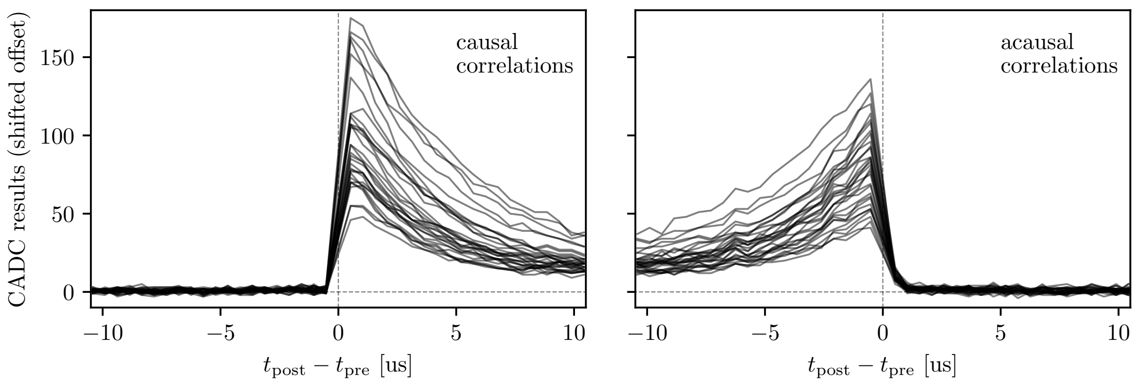


Figure 3.20: Correlation recordings of 31 synapses belonging to one neuron on HICANN-DLSv3. Every synapse was exposed to a number of pre- and postsynaptic spike pairs each with fixed inter-spike interval $\Delta t = t_{\text{post}} - t_{\text{pre}}$ prior to CADC readout. The left plot shows the voltages collected for the causal correlation branch, the right plot for the acausal branch. One synapse was blacklisted due to overly bad performance within this measurement. The data was also recorded by Johannes Weis within the scope of his Bachelor's thesis. Further details can be found in Weis (2018).

Correlation readout

As mentioned, the main purpose of the CADC is the conversion of the synaptic correlation measurements that are relevant for numerous learning rules, like, for example, STDP. Figure 3.20 shows the results of such measurements for both correlation branches of a column of synapses on HICANN-DLSv3.

Membrane voltage readout

Besides synaptic correlation measurements, the CADC can also digitize the membrane potentials of the neurons. For this purpose, digital switches in the synapse array and the neuron readout multiplexer can be configured such that an amplifier located at the neuron sends the buffered membrane potential to one of the synaptic correlation readout lines. This feature has recently been used by Cramer et al. (2020a) to train a three-layer network on HICANN-X to classify handwritten digits from the MNIST dataset (LeCun et al., 1998). The underlying training algorithm was proposed by Zenke and Ganguli (2018) and requires knowledge of the membrane potentials to compute the backpropagation signals. For an elaborate description of the algorithm and the technical details, the reader is kindly referred to Cramer et al. (2020a). Figure 3.21 shows the recorded membrane traces and spike activity during one classification cycle. The CADC recorded the membrane potentials of 128 neurons simultaneously at a rate of 400 kHz. The rate is adequate for the experiment and naturally results from the execution time for serial data access and arithmetic in the PPU.

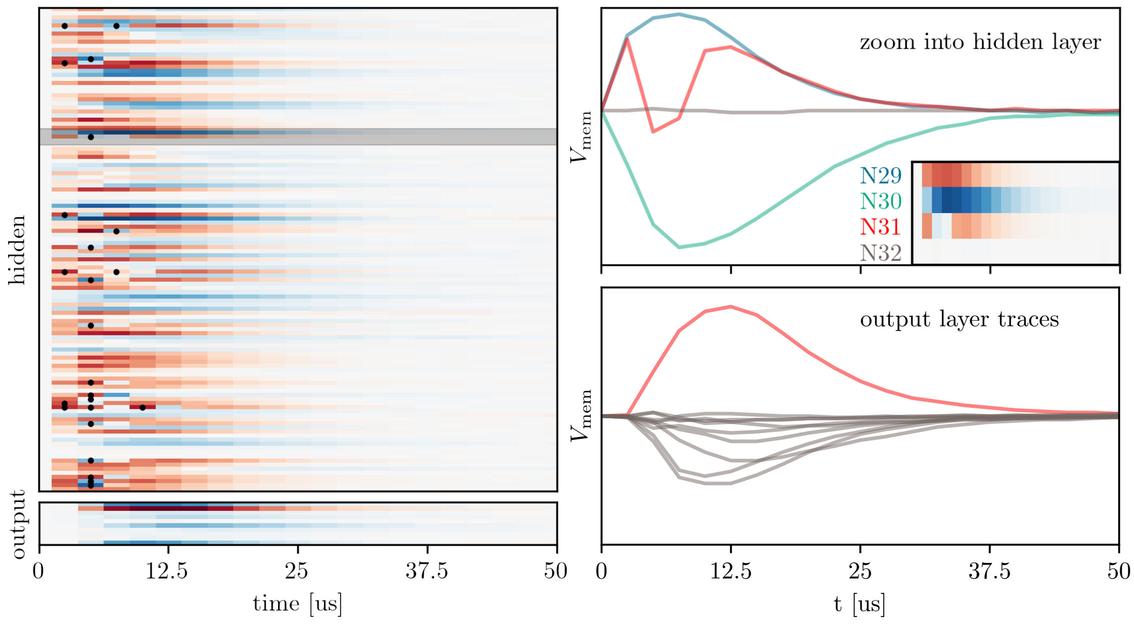


Figure 3.21: Parallel recording of membrane traces with the CADC. The left side shows the color-encoded membrane potentials of 118 neurons in the hidden layer and 10 neurons in the output layer of a spike-based classifying network. The black dots indicate spike events of the respective neurons. At $t = 0$, an MNIST-digit is presented to the input layer (not part of the above illustration), which causes postsynaptic potentials in the neurons of the depicted hidden layer, which, in turn, cause output spikes that ultimately reach the output layer. The output layer neurons are configured as non-spiking leaky integrators, such that the classification result can be obtained from their membrane potentials. The membrane potentials of a subset of hidden layer neurons (highlighted in gray) are shown in more detail in the top right panel. The zooming region covers the traces of four neurons (29 to 32) whose membrane potentials are shown in the main plots. The small embedded figure is merely an enlarged view of the zoomed region with the same color encoding as the left plots. The lower plot on the right-hand side shows all membrane traces of the output layer. Most membrane potentials are either inhibited or unaffected, while one trace clearly separates from the others (red). The index of this most active neuron identifies the classification response of the network. The recordings where kindly provided to the author by Sebastian Billaudelle. An elaborate description of the underlying experiment can be found in Cramer et al. (2020a).

3.5.4 Discussion and outlook

The CADC calibration and accuracy updates state major improvements as compared to the previous design. Given the area and power constraints, the component probably operates close to the achievable optimum, which is supported by the above mentioned comparative study by Sundstrom et al. (2008) (see figure 3.19).

However, the most limiting factor is perhaps the comparably low sampling rate of $\mathcal{O}(1-2\mathrm{MHz})$, which is mostly due to the ramp-compare architecture that intrinsically requires at least 256 clock cycles for an 8 bit conversion. One possible way to overcome this is to switch to a SAR-architecture instead. This ADC type requires the same amount of clock cycles per conversion as the desired number of bits, i.e., 8 cycles in this case. Since SAR-ADCs are among the most power-efficient architectures, this could also state a further improvement in terms of energy-efficiency (Bashir et al., 2016). However, these two advantages have to be be paid for predominantly by three disadvantages.

First, SAR-ADCs are prone to differential nonlinearities (DNL). This means that the difference between two adjacent analog values does not map to the same difference in the converted digital domain. For example, when the input voltage crosses the equivalent of one LSB, a jump of multiple LSBs in the digital domain could be the consequence in an ADC with a bad DNL. At another absolute input voltage the same differential increase could result in no or even a negative change in the converted value. Thus, while the overall input range could map well to its output range (a good INL), the results could still be compromised by differential nonlinearities (DNL).

Especially for learning algorithms in neural networks, this can state a significant problem, since they often rely on differential instead of absolute variables. Usually, this type of nonlinearity is compensated for by larger capacitor sizes (mismatch reduction) and occasionally by calibration circuits. Both options, however, are facing substantial technical difficulties in the current implementation due to the tight area constraints. Ramp-compare ADCs, in contrast, are intrinsically robust against differential nonlinearities.

Second, the size of an SAR-ADCs grows exponentially with the number of bits N, if not pipelined. I.e., the area that is required for the SAR capacitor array is proportional to $2 \cdot 2^N \cdot A_{\mathrm{min}}$, where A_{min} is the size of the smallest capacitor. Using serially connected capacitors for the lowest bits, the scaling can be reduced to $2 \cdot (2^{N-2} + 2 + 2^4) \cdot A_{\mathrm{min}}$. Because of the necessary mismatch reduction, A_{min} should be on the order of at least $5\,\mathrm{\mu m}^2$ net. The necessary area is then roughly $700\,\mathrm{\mu m}^2$ only for the SAR capacitor bank. Therefore, an SAR-ADC would likely triple the area requirements as compared to the presented ramp architecture. However, a three-fold increase in area is possibly still within a manageable frame on the current chip. Moreover, when transitioning to larger die sizes (e.g. for wafer-scale integration), this problem can potentially become entirely negligible.

Third, the input capacitance of an SAR-ADC is a few orders of magnitude higher than that of a typical ramp-ADC. In practice, this can decrease the accuracy significantly because the sampled signal is connected to the low-impedant ADC input. As the CADC multiplexes the synaptic rows, frequently varying input voltages have to be expected, worsening the problem. This obstacle can only be resolved by reducing the impedance of the signals to be measured, which automatically necessitates higher current consumption in the corresponding buffer circuits. In the MADC readout chain[1], the same issue causes the readout amplifier to require $\mathcal{O}(1\,\mathrm{mA})$ supply current, which is comparable to the current consumption of the MADC itself.

Taken together, the three disadvantages require certain trade-offs to be accepted for the SAR approach. Nevertheless, especially in the machine learning mode (see section 3.3.1), where the membrane potentials represent the results of vector-matrix multiplications, the CADC sampling rate is potentially one of the speed-limiting factors. The other bottlenecks are currently the integration time on the membrane capacitor and the reset time of the neuron. A thorough analysis of the performance in machine learning applications is subject of currently ongoing research. Therefore, the SAR architecture might indeed be of interest in future designs and should be kept in mind.

The currently implemented ramp-compare CADC, however, combines a lot of advantages into a readily usable, scalable, and robust solution. Concerning the speed disadvantage, there also exists a potential improvement that could be implemented without much effort in the digital CADC back-end. I.e., the CADC conversion time can be shortened by making the maximum counter value configurable, trading digital resolution for conversion speed. Limited by the bandwidth of the ramp generator and the time the ramp reset requires, the maximum sampling rate can likely be increased by a factor of 5 to $\sim$8.8 MHz, when a resolution of 5 bit is still acceptable. A sampling rate of 5.6 MHz could be achieved at a 6 bit resolution. Considering the noise on the synaptic input line, this resolution might be well sufficient for many tasks.

[1]The MADC is an SAR-ADC.

4 Accelerated mechanics

"All three crystals were housed in the basement now, just centimetres away from the Play Pen: a vacuum chamber containing an atomic force microscope with fifty thousand independently movable tips, arrays of solid-state lasers and photodetectors, and thousands of micro-wells stocked with samples of all the stable chemical elements. The time lag between Sapphire and this machine had to be as short as possible, in order for the Phites to be able to conduct experiments in real-world physics while their own world was running at full speed."

– Greg Egan, Crystal nights, 2009

4.1 Motivation

In his science fiction short story *Crystal nights*, Greg Egan describes how an artificial, accelerated world, called *Sapphire*, is simulated on an extremely powerful supercomputer. Within Sapphire virtual beings, the *Phites*, are exposed to an evolutionary pressure that requires them to evolve higher and higher intelligence. After developing space travel, the Phites find a black monolith on their moon that contains interfaces to a machine that is not part of their simulation but of the outside physical world, the *Play Pen*.

Since Sapphire, like the neurons on BrainScaleS-2, runs accelerated with respect to the outside world, the problem that the engineers in Egan's story face, is similar to the problem that this section is about. I.e., constructing a mechanical robotic interface for an accelerated intelligent system.

More generally, this chapter can be seen as a contribution to the field of neurorobotics, which is mainly concerned with the neuronal control of physical or virtual agents. Within a large part of the research in this field, the neural control organs are implemented in the form of digitally executed simulations on conventional computer architectures. However, there are some isolated projects that use neuromorphic hardware instead. For example, Massoud and Horiuchi (2010) presented a ring attractor network that can track an angular direction on a custom neuromorphic ASIC. These networks are similar in function to some brain areas which are commonly found in various animal species, from mammals to insects (see chapter 5). More recently, a similar circuit was presented with further extensions that allow the encoded direction to be updated precisely by means of differential movement inputs (Kreiser et al., 2018b,a, 2019a,b). In Kreiser et al. (2020), this network was successfully employed to estimate the head pose of a small humanoid robot. Zhao et al. (2020) derived a neural motor controller from a similar circuit to control the head rotation of the same robot. Cheng et al. (2020) recently presented another agent who is able to intercept moving objects based on the digital neuromorphic SpiNNaker system (Furber et al., 2014). As already mentioned in section 1.1, none of the above examples runs significantly faster then biological real-time though. Moreover, none of them employs the neuromorphic substrate for much more than a fractional aspect of the agents.

The motivation behind the presented robot is to study how accelerated neuromorphic nervous systems can cybernetically interact with external environments, while the interfaces between the two worlds are realized by a set of biomimetic actuators and sensors. I.e., for the presented robot, the sensors, actuators, and the agent are fully neuromorphic. In this way, the accelerated neural network experiences a similar state to biological nervous systems that engage in a constant interplay of action and reaction. Potentially, this should facilitate insight into many dynamical network effects like the control of motion sequences or fast information propagation through the network. Moreover, as the neuromorphic agent is able to determine the sensed content on its own, this experiment architecture also enables studying phenomena like attention.

This chapter first outlines the physical considerations that preceded the technical concepts. Then the development and characteristics of a prototype are described, followed by those of the final robot. For both incarnations, experiments are presented that reflect the biological plausibilities and capabilities of the systems.

Contribution

Louis Jussios selected and developed the rotational position sensors for the first robotic prototype (section 4.3.2) and Malte Prinzler and Sebastian Billaudelle contributed to the mechanical construction of it. Furthermore, Malte Prinzler contributed to some electronic components and conducted the first nontrivial experiment on this prototype which is outlined in section 4.5. All other parts of this chapter were conceptualized, implemented, and evaluated by the author of the thesis.

4.2 Requirements and limitations

The intrinsic speed-up of BrainScaleS-2 represents both a challenge and an opportunity for neuromorphic robotic experiments. On the one hand, the demands that have to be met by a mechanical system that is supposed to move $1000\times$ faster than biological systems are significant and hard to satisfy with of-the-shelf components. On the other hand, the increase in experiment throughput is high enough to leverage robotic optimization schemes to a new qualitative level. I.e., costly optimization algorithms, like evolutionary strategies, can be executed within days instead of years when implemented serially.

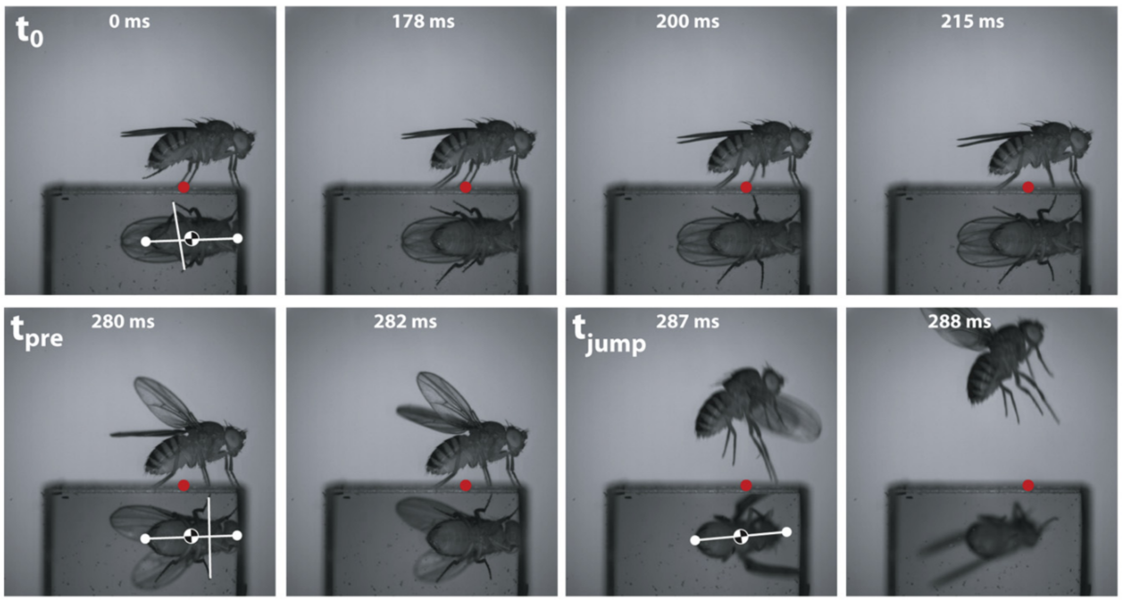


Figure 4.1: *Drosophila melanogaster*'s physical reactions after being confronted with the visual cue of a fly swatter at $t_0 = 0$. The first reaction becomes visible around 200 ms after the cue. Reprinted from Gwyneth Card, and Michael H. Dickinson, *Visually mediated motor planning in the escape response of Drosophila*, Current Biology 18(17) (2008): 1300-1307, Copyright (2008), with permission from Elsevier. (Card and Dickinson, 2018).

Except for optimization advantages, the time scales of the actions that BrainScaleS-2 could take as a neural robotic controller are on the order of a few microseconds. It could therefore be used in high-speed applications, like for example micro-sized or very fast aerial vehicles, motor control circuits, or ultra sound and radar applications.

In order to develop an intuition about the demands that the BrainScaleS-2 speed-up states for a physical system, first a look is taken at neural actuator control in biological systems.

4.2.1 Actuators

The speed, or more precisely the acceleration, of movement of biological actuators with nervous systems is in a certain proportion to their size (Greenewalt, 1962; Healy et al., 2013), which among other reasons is simply due to the inertia of their extremities. For example, the largest animal on our planet, the blue whale, moves with calm, graceful strokes that typically span several seconds. In contrast, some midges flap their wings up to 1000 times per second (Sotavalta, 1953). This frequency, however, is beyond the reach of neural dynamics (see section 2.1.2) and relies on a slick interplay of mechanical stretch and motor control instead.

As the typical spiking frequencies of neurons lie around 10 Hz to 100 Hz, the relevant lower end of motion time scales that are accessible by neural control should typically be on the order of 10 ms to 100 ms. Support for this range is provided by experiments with *Drosophila melanogaster*. In Card and Dickinson (2018), a high-speed camera recorded the motion of fruit flies when confronted with a fly swatter (see figure 4.1). It takes approximately 200 ms for the fly to process the visual input and initiate motor action. The leg movements and wing flapping then happens within a few 10 ms.

Considering the BrainScaleS-2 speed-up, we therefore desire a mechanical system that is capable of performing changes in motion on the order of 10 µs to 100 µs. For a robotic application, these small timescales are crucial, since longer time scales can usually easily be achieved by reducing the force of the actuators or simply by adding mass. That is to say, slower motion is always accessible.

To estimate the feasibility of such a system, first the energy E that is required to translate an initially resting mass m over a distance x within a time interval τ is considered:

$$E = 2 \cdot m \cdot \left(\frac{x}{\tau}\right)^2 . \tag{4.1}$$

For a constant acceleration, this implies a necessary power of

$$P = 2 \cdot m \cdot \frac{x^2}{\tau^3} . \tag{4.2}$$

Assuming that electronic sensors and mounting accessory might be attached to the actuator, a mass of a few gram is hardly avoidable. Moreover, for movements on biological length scales, x should be on the order of perhaps 1 mm. Thus, the power required to move an initially resting sensor with a mass $m = 3\,\mathrm{g}$ over a distance $x = 1\,\mathrm{mm}$ within $\tau = 10\,\mu\mathrm{s}$ is $P = 6\,\mathrm{MW}$, and therefore already on a scale that lies beyond the capacities of a typical laboratory environment (at least if driven continuously). Fortunately, due to the inverse cubic dependence on τ, the power drops by three orders of magnitude to $P = 6\,\mathrm{kW}$ if the desired mechanical response time is decreased by only one order of magnitude to $\tau = 100\,\mu\mathrm{s}$. If it is decreased further to $\tau = 500\,\mu\mathrm{s}$, that is half a second in biological time, the power consumption drops to only 48 W, which can easily be satisfied by standard low-cost power supplies. The resulting acceleration is then on the order of

$$a = 2 \cdot \frac{1\,\mathrm{mm}}{(500\,\mu\mathrm{s})^2} = 8000\,\mathrm{m\,s^{-2}} . \tag{4.3}$$

Although, this response time is not in the desired regime of 10 µs to 100 µs, it is still within the range of just a few spikes. Also, the quadratic dependency on the distance x allows for dividing the response time by a factor of 4 to $\tau = 125\,\mu\mathrm{s}$, if a minimum movement of 0.5 mm is accepted instead.

The next aspect to look at are the spike-actuator transfer functions, i.e. the muscle force with respect to the neural input commands. Figure 4.2 shows a simplified schematic of vertebrate motor units. Each motor unit is comprised of a motor neuron and a set of muscle fibers. The force response at various input spike rates of such a unit is illustrated in figure 4.3. Here, the relevant spike rates are on the order

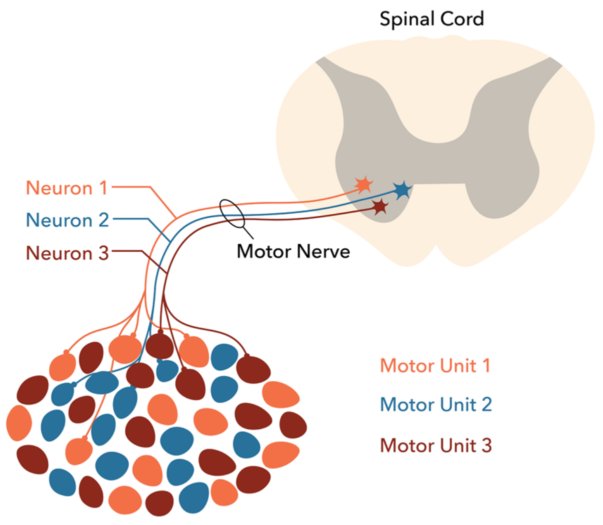


Figure 4.2: Schematic of biological motor units. A motor unit consists of a motor nucleus in the spinal cord, an axonal connection to a muscle, and some muscle fibers. Figure taken from Walsh and Sved (2019).

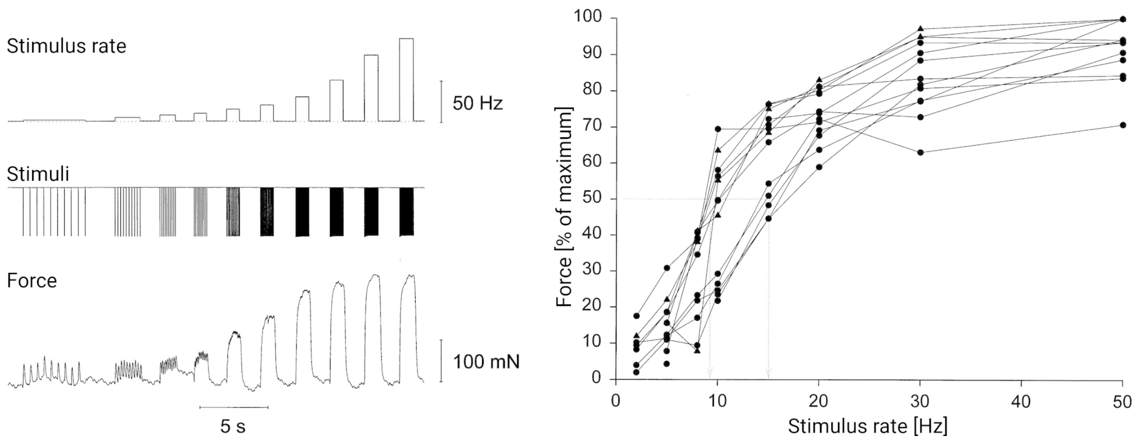


Figure 4.3: Force response of motor units related to finger movement with respect to a varying spike rate stimulation. Results were obtained by intraneural motor axon stimulation in humans. Adapted from Fuglevand et al. (1999) with permission of The American Physiological Society.

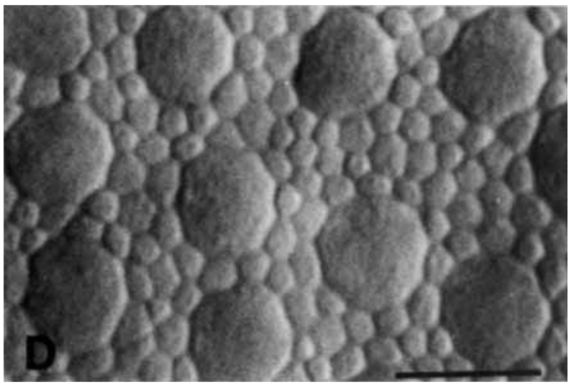


Figure 4.4: Human retina with rods (small) and cones (large). Reference scale is 10 µm. Taken from Curcio et al. (1990). Reprinted with permission from John Wiley and Sons and Copyright Clearance Center. Copyright (1990) Wiley - Liss, Inc.

of 0 Hz to 50 Hz with a typical saturation at around 30 Hz. All motor units that constitute an entire muscle are referred to as a muscle pool (Kandel et al., 2013). When a muscle becomes active, small motor units within this pool become active first. As more force is required, the more and the larger motor units get activated by spinal interneurons. This process, also known as motor unit recruitment (Kernell, 2006; Kandel et al., 2013), allows for high precision at tasks that require weak forces, and strong support of large units, when strength is more important than precision. In that way, the nonlinear force response of a single motor unit averages out and a more homogeneous, linear transfer function for the entire muscle with respect to the interneuron input is achieved.

One possible medium to realize artificial muscles that operate within the above mentioned power scales, are electromagnetic actuators. Since these devices usually have a very homogeneous, within a certain range even linear, force response with respect to the input current, the recruitment of multiple units is not necessary and the force can instead straight-forwardly be controlled by the overall input current. In accordance with biology, the input spike rate for the current control circuits of the accelerated robotic actuators is therefore desired to cover a maximum range of 0 kHz to 100 kHz.

4.2.2 Sensors

Nervous systems acquire information on the physical system and environment that they are embedded into, via a large variety of sensory cells. Typically, sensor cells rely on specialized physicochemical processes that induce changes in the cells' membrane potential when stimulated by extracellular information, e.g., light, pressure, heat, sound, etc. (Kandel et al., 2013). The changes in the membrane potentials eventually cause action potentials that are transmitted as spikes into the nervous system. Since so many different types and subtypes of sensor cells exist, only a few examples are presented here in order to not leave the scope of this manuscript. Two aspect are of particular interest for neuromorphic sensors: the typical encoding schemes, and the response dynamics.

Sensor cells typically appear where they are useful for the organism that carries them (see section 2.1.4). Their spatial density is usually proportional to their usefulness at the given location. Since every sensor has a certain energy consumption, they are of course placed economically. For example, mechanoreceptive cells are very densely packed at the tongue and lips, as this allows the organism to detect hazardous material before swallowing it. Also our finger tips contain a lot of touch receptors, since this enables a high level of control and precision of movement, which, in turn, is useful for object manipulation, tool building and hazard detection. On the other hand, areas like the trunk, do only contain as many receptors as are necessary to detect relevant outside influences.

An area that is extremely densely packed with sensor cells is the retina (see figure 4.4). It is filled with light sensitive cells, called rods, and color specific cells, called cones. The rods reach a density of about up to 1 cell every 2 µm at the regions highest resolution. In front of this layer of rods and cones is a layer of various types of retinal neurons, that implement some rudimentary receptive fields over the light receptive cells. Ultimately, retinal ganglion cells represent the output of these fields and transfer the information via long axonic connections into the direction of the central nervous system (Hubel, 1995). For humans, this bundle of axons, the optic nerve, contains about 1 158 000 nerve fibers on average (Jonas et al., 1992). Thus, assuming an average firing rate of 10 Hz, the typical data rate of this bundle is on the order of 10 million spikes per second. Translated to the BrainScaleS-2 speed-up and encoding, this would correspond to a data rate of about $30\,\mathrm{GB\,s^{-1}}$. While future scaled-up versions of BrainScaleS-2 might potentially be able to process such an amount of data, it is by far too large for the current single chip setups. Moreover, after arriving in the brain, the optical information is distributed to and processed by the striate area (a part of the visual cortex), that contains a number of neurons that is orders of magnitude larger than the number of axons within the optic nerve. Figure 4.5 shows how the optical information is distributed over the striate cortex of macaque monkeys.

Since a biologically plausible neuromorphic visual cortex should behave in the same way, even the limitations of fully upscaled wafer version would be surpassed. Therefore, reducing the number of axons and, hence, the sensor dimensionality is inevitable for these kinds of experiments within the foreseeable future. Since the BrainScaleS-2 prototype chip on which the robotic experiments within this thesis were conducted, contains only 32 neurons, the sensor dimensionality has to be limited even more drastically to only 1 to 10 sensor channels in total.

Apart from the sensor dimensionality, another interesting aspect is the way in which a single sensor unit transforms a scalar stimulus into a spike train. Figure 4.6 shows how slowly (A) and rapidly (B) adapting mechanoreceptive cells respond to pressure stimuli in terms of their output spikes. In column A, the most apparent observation is that the overall firing rate is in some proportion to the amplitude of the applied stimulus. Another observation is that the cells adapt to the constant stimuli by reducing their output rate over time. This behavior is even more pronounced in B, where cells were stimulated with rising signals. These rapidly adapting receptors fire only when the stimulus is changing over time and are insensitive to the absolute amplitudes. This dynamical response is not exclusive for mechanoreceptors but

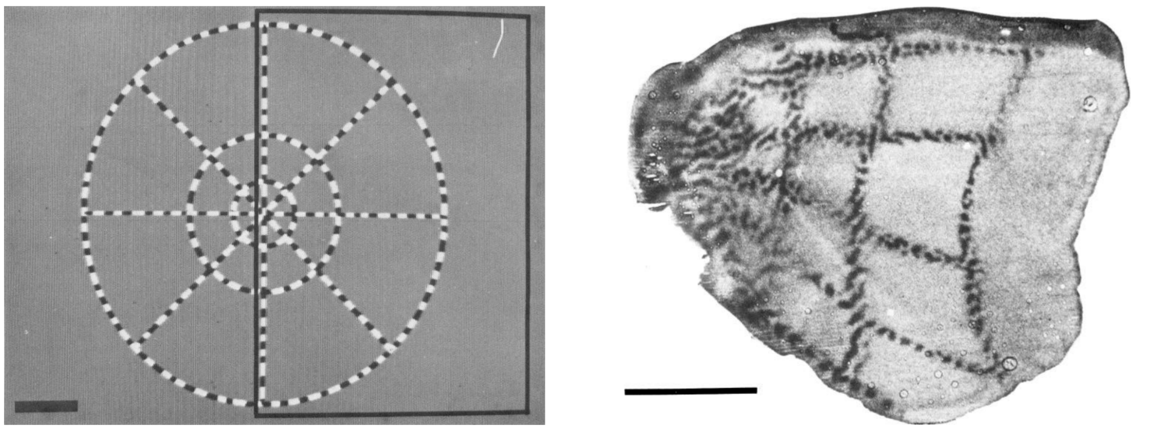


Figure 4.5: The spatial representation of visual information in the striate cortex of macaque monkeys. For this experiment, a monkey was anesthetized and medically paralyzed in order to suppress eye movements. A static visual pattern was then shown in front of it such that the pattern was projected onto its retinas for approximately 25 min to 30 min before the monkey was euthanized. The striate cortex was then removed and the activity visualized on an autoradiograph. At the beginning of the experiment the monkey was injected radioactive 2-deoxy-d-glucose that is metabolized by neurons. The more active neurons are, the more energy they require and the more metabolism takes places. Therefore, high neural activity is correlated with high concentration of this special radioactive glucose. Left: Target-shaped visual pattern that is presented to the monkey. For maximum neural response, the black and white checks alternatingly flickered with a frequency of 3 Hz. The black border around the right halve of the pattern indicates the part of the field of vision the neural response belongs to. the scale bar denotes 2° in the visual field. Right: The autoradiograph of the fourth layer of the striate cortex. The structure of the presented pattern is reproduced on the cortex, proving a topological conservation of visual information. The left side corresponds to the center of the pattern, the right side to its outer regions. The scale bar length is 1 cm. The number of neurons in this area is about 78 000 mm^{-2} for macaque monkeys (Beaulieu and Colonnier, 1989). Taken from Tootell et al. (1988). Copyright 1988 Society for Neuroscience.

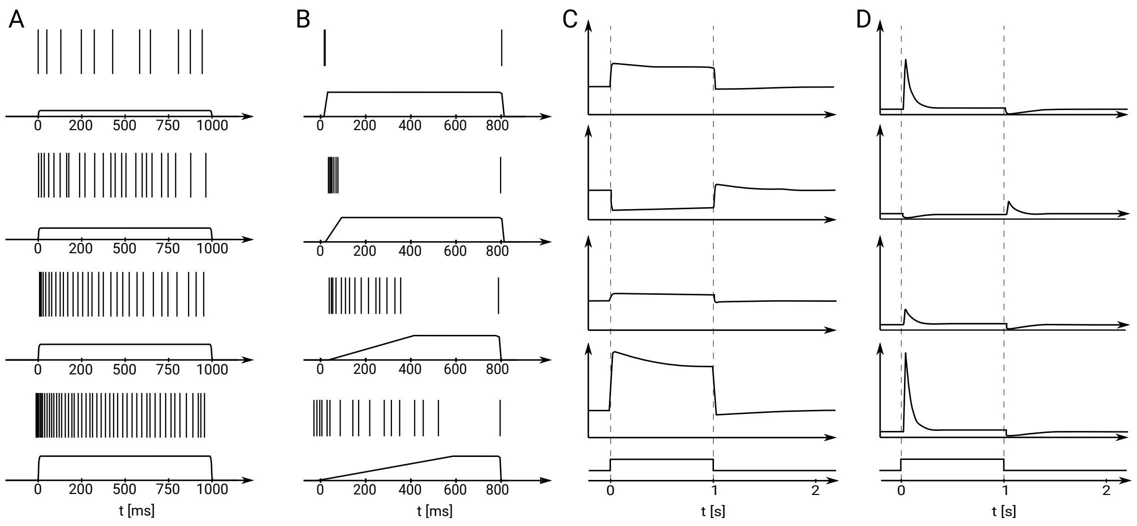


Figure 4.6: A, B) Spike response of a few mechanoreceptive sensory neurons (vertical bars). The stimuli over time are given below each spike trace. All cells start spiking when the stimulus is being applied and adapt to it by decreasing their output activity over time. Some receptors adapt slowly (A) and thereby indicate mostly the stimulus amplitude, others adapt rapidly (B) being mostly sensitive to changes in the applied stimuli. C, D) Spike rate response of slowly (C) and rapidly (D) adapting retinal ganglion cells. Each column shows four typical response functions and the applied stimulus in the bottom line, which is switched on at $t = 0$ and switched off again at $t = 1\,\text{s}$. Sustained cells (C) respond by a change of a certain baseline activity as long as a stimulus is applied. In transient cells (D), the activity is influenced by the change of the stimulus signal but decays back to the baseline level within a few 100 ms. Note that also sustained cells may have a certain amount of transient decay. The data has been extracted from Mountcastle et al. (1966), Talbot et al. (1968), and Kandel et al. (2013).

appears in many other sensor cells as well (Hubel, 1995; Kandel et al., 2013).

Figure 4.6 also shows a similar scheme for slowly (C) and rapidly (D) adapting retinal ganglion cells. In this case, the stimulus is increased relative to some baseline activity instead of starting from and dropping back to zero as in A and B. When the stimulus is removed, these cells undershoot. The second row reveals inverse output characteristics, i.e., these cells decrease their activity upon receiving a positive stimulus.

The advantage of this adaptation scheme is that it greatly extends the dynamic range of perception by conveying relative instead of absolute information. For example, our ears are able to detect pressure variations within a dynamic range of up to 120 dB (Zwicker and Fastl, 2013). Another example is our eyes, that can see under night sky luminance levels of only $10^{-3}\,\mathrm{cd\,m^{-2}}$ as well as in bright daylight at $10^{6}\,\mathrm{cd\,m^{-2}}$, i.e. within a range of nine orders of magnitude. At any such level of adaptation, they can recognize details that vary by $1 : 10^{4}$ (Banterle et al., 2017).

If the encoding happened in direct proportion to the incoming light intensity with a maximum spiking rate of 100 Hz, the maximum jitter of the individual spikes would have to be on the order of 1 µs for the detail recognition and on the order 10 ps if one sensor was supposed to capture the entire dynamic range with that precision[1] Due to adaptation, this becomes unnecessary, as most of the sensors auto-adjust to the current offset conditions. Saccades further help scooping the full potential of this differential encoding scheme, since they cause frequent changes in the light patterns that are mapped on the retina.

Seen from a systems theory perspective, the response functions in figure 4.6 appear to contain proportional, as well as high-pass filtered stimulus components. I.e., similar response functions in the form of an output rate r depending on a time-dependent continuous input stimulus $s(t)$ can be constructed as a superposition of a proportional, a differential, and constant term:

$$r(t) = a \cdot s(t) + b \cdot \frac{d}{dt}s(t) + c \tag{4.4}$$

The next section describes how these presented biological de- and encoding schemes can be mapped to a high-speed robotic application.

[1]Neglected is the fact that the eye's apertures also contribute to the adaptation. Given that they can modulate the amount of incoming photons by a factor of roughly 10 to 100, one or two orders of magnitude can be subtracted from the total range. The total dynamic range of the retina therefore still stretches over seven to eight orders of magnitude. A similar mechanism also exists in the ears. Here, the *stapedius* muscle contracts in a way that dampens the mechanical signal that is guided to the *cochlea*, when loudness increases.

4.3 PlayPen

The main goal was to achieve a possibly flexible system, that simultaneously satisfies the tight timing constraints imposed by the 1000-fold speed-up of the neural network. Since this endeavor of interfacing an accelerated neuromorphic agent with the real world uncannily resembled the situation in Egan's short story, the name *PlayPen* was chosen for this project.

4.3.1 Actuators

As the research focus was on the neuromorphic cybernetics rather than on the robot itself, the building and design effort was to be kept possibly low. The high mechanical acceleration already excludes something like a surface-bound vehicle, as the traction of wheels would require additional pressure of some kind to mediate enough force. Robot-sumo competitions (Liu and Zhang, 2004; Erdem, 2007; Wikipedia, 2019) provide examples for very quickly accelerating surface-bound robots. In these competitions, two robots are put into a circular arena with the goal for each robot to push the other one out of the ring. In order to gain maximal traction, these robots are being pulled to the metal arenas by strong neodymium magnets. A match can end within fractions of a second. Judging from competition videos (McGregor, 2019) (for a lack of scientific publications that reflect the state-of-the-art in robot-sumo), the robots can traverse an arena with a diameter of $\approx 1\,\mathrm{m}$ within $100\,\mathrm{ms}$ to $200\,\mathrm{ms}$ starting from and ending at resting state. Thus, they reach accelerations of approximately $100\,\mathrm{m\,s^{-2}}$ to $400\,\mathrm{m\,s^{-2}}$. Although this acceleration is remarkably high for surface-bound vehicles, it is still more than an order of magnitude below what is required for BrainScaleS-2 (see equation (4.3)).

Higher accelerations can be achieved by mediating the force through, for example, bearings or joints instead of friction.

A recent example of very fast lightweight robotic actuators is the wings of flying microbots (Chen et al., 2019). Driven in resonance, these wings can reach a displacement of $\approx 1\,\mathrm{mm}$ at a flapping frequency of $500\,\mathrm{Hz}$. Roughly, this corresponds to an acceleration of $4000\,\mathrm{m\,s^{-2}}$, which is only a factor of two away from equation (4.3). Being the result of very recent research in micro robotics, actuators like that are after all not available of the shelf yet. Neither are they suitable for carrying sensors, as their own weight is only about $100\,\mathrm{mg}$ which would be greatly exceeded by PCB-based sensors.

Another recently published high-speed actuator is a piezoelectrically driven Delta robot (McClintock et al., 2018). Delta robots are actuators that can access three degrees of translational movement by using three motors in a pantograph-like manner. According to the authors, this robot's bandwidth is 15 to 25 times higher than that of previously available Delta robots. However, the maximum accelerations are on the order of $215\,\mathrm{m\,s^{-2}}$, which is again far below what equation (4.3) demands. Since these actuators are part of recent academic research as well, they are neither readily available, nor reproducible without the risk of distorting the neuromorphic research focus.

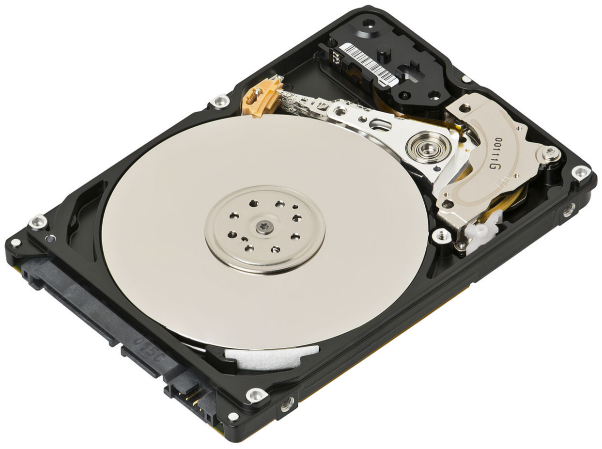


Figure 4.7: Hard-disk drive (HDD) with magnetic storage disks and voice coil actuator. Image taken from Evan-Amos (2013).

Very common everyday objects that have exceptionally fast actuators are hard-disk drives (HDDs), though.

In HDDs, the magnetic information on the rotating disks is accessed by a read and write head that is brought into position by a moving coil motor (see figure 4.7). This *voice coil* is rotatably mounted such that the coil at its rear part is penetrated by a magnetic field $\mathbf{B}$. When a current $\mathbf{I}$ flows through the coil, the Laplace force

$$\mathbf{F}_{\text{side}} = n \cdot l \cdot \mathbf{I} \times \mathbf{B} \tag{4.5}$$

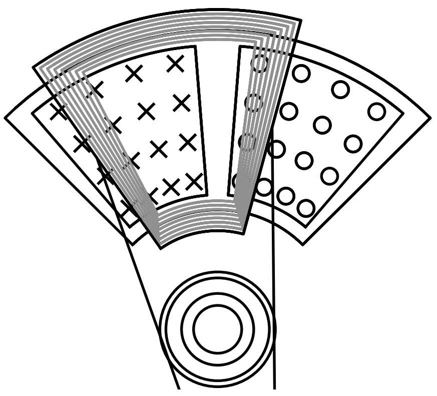
is induced on one side of the coil. Here, n is the winding number and l is the length of the coil segment that is penetrated by the magnetic field. As the other side of the coil is immersed in a magnetic field of same magnitude but opposite direction with inversely flowing current, the net force acting on the voice coil is $\mathbf{F} = 2 \cdot \mathbf{F}_{\text{side}}$ Since n, l, and $\mathbf{B}$ are constant and fixed for a given voice coil, and since $\mathbf{I}$ is always constrained to be perpendicular to $\mathbf{B}$, the net force can be written as

$$F = c_{\text{coil}} \cdot I. \tag{4.6}$$

The accelerating force should, therefore, be proportional to the coil current I. Figure 4.8 offers further support for equation (4.6) by measurements of the voice coil torque at varying coil currents.

The accelerations that HDD-voice coils can achieve are on the order of what is desired from equation (4.3). For example, assuming that a the read/write head travels over half of the disk width, i.e. over $\approx 1\,\text{cm}$, to access random data and given an average seek time of 3.6 ms (Wes, 2010), the acceleration should be on the order of at least $3000\,\text{m}\,\text{s}^{-2}$.

HDD-voice coil actuators are therefore well suited for simple robotic experiments at BrainScaleS-2 speed-up. Constrained by this selection, two voice coil actuators were combined into a pantographic robot. The basic idea is sketched in figure 4.9. Note that, additional levers and moving parts decrease the acceleration of a naked HDD-voice coil. To a certain extend this can be compensated for by increasing the coil current but an effective lower maximum acceleration is likely to be the case



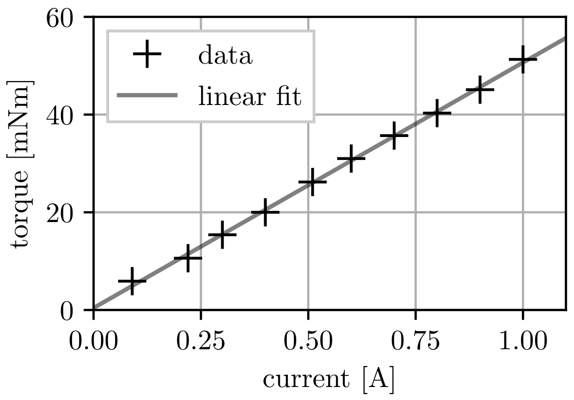


Figure 4.8: Left: Schematic of the voice coil within the magnetic field spawned by the statically mounted permanent magnets. The magnetic field lines on the left hand side (indicated by crosses) enter the plane, while the field lines on the right hand side exit the plane (rings). The current can either flow clockwise or counterclockwise through the coil, causing the actuator to move clockwise or counterclockwise, respectively, due to the Laplace force. Right: The measured torque of a voice coil actuator vs. coil current and a linear fit. The measurements were conceptualized and recorded by Malte Prinzler (Prinzler, 2018).

in the real setup. For now, the maximum achievable acceleration is optimistically assumed to be on the order of

$$a_{\max} \approx 3000\,\mathrm{m\,s^{-2}}. \tag{4.7}$$

The maximum frequency of an oscillation over a range of 1 mm is therefore on the order of

$$f_{\max} = \frac{1}{2\pi}\sqrt{\frac{a_{\max}}{x}} = \frac{1}{2\pi}\sqrt{\frac{3000\,\mathrm{m\,s^{-2}}}{0.5\,\mathrm{mm}}} = 390\,\mathrm{Hz}. \tag{4.8}$$

Next, the signal processing and amplification circuits are discussed that together with the actuators form a robotic motor unit.

To translate neural activity into the current that drives the coil, a multi-stage circuit is implemented that converts individual spikes that come from the Spike I/O of the BrainScaleS-2 prototype system into amplified current pulses. Spikes leave the neuromorphic system as voltage pulses with a width of 500 ns. Due implementation details on the Flyspi FPGA board, the voltage amplitudes are either 2.5 V, 3.3 V, or 5.0 V, depending on the sender neuron index (see also section 3.2). A battery of `TXB0104/8PW` level shifters (Tex, 2018b,c), therefore, first unifies all voltage levels to 5.0 V before they enter the next circuits (see appendix C.4 Sheet 1/3).

To cause a significant actuator response, the next stage extends the pulse length. Figure 4.10 shows the schematic diagram of one such unit that is also referred to as *pulse shaper*. The 500 ns-pulse charges a capacitance C over a small signal diode D, such that the charge cannot flow back, when the input voltage level goes back to 0 V.

Instead, the capacitor discharges mainly over a variable resistor R which, together with the capacitor, forms an RC-circuit with exponential decay characteristics. The decay time constant $R \cdot C = \tau_{\mathrm{RC}}$ determines the shaped pulse length. The voltage across C, V_{C}, is applied to the input of an inverting `74HC14D` Schmitt trigger (Nex, 2020).

Schmitt triggers are digital buffering ICs with a certain level of input hysteresis. I.e., if the output of the Schmitt inverter is `HIGH`, it takes an input voltage $V_{\mathrm{in}} \geq V_{\mathrm{TP}}$ to flip the output signal to `LOW`. As soon as the output is `LOW`, it takes an input voltage of $V_{\mathrm{in}} \leq V_{\mathrm{TN}} < V_{\mathrm{TP}}$ to switch the output back to `HIGH` again. The difference of the two threshold voltages, the hysteresis voltage $V_{\mathrm{H}} = V_{\mathrm{TP}} - V_{\mathrm{TN}}$, is particularly helpful when noisy or slowly rising input signals are to be translated into faster digital signals. Without hysteresis, the output level of a standard inverter with a single input threshold V_T would quickly switch between `HIGH` and `LOW` during a slow transition from $V_{in} < V_T$ to $V_{in} > V_T$ if the signal is noisy.

In the case at hand, the hysteresis allows for creating a `LOW` signal on the output while C is being discharged from its initial voltage $V_{\mathrm{C}}(t \approx t_{\mathrm{spike}}) \geq V_{\mathrm{TP}}$ to $V_{\mathrm{C}}(t \approx t_{\mathrm{spike}} + \tau_{pulse}) \leq V_{\mathrm{TN}}$. This negative pulse is then inverted by a second Schmitt inverter such that an extended positive output pulse leaves the pulse shaper. By tuning R and C, the pulse time is usually set to $\tau_{pulse} = 8\,\mu\mathrm{s}$.

The next step is to turn the shaped voltage pulse into an amplified current pulse that can drive the voice coil. The basic idea is to drive each terminal of a coil by a voltage amplifier with low output impedance and current limitation. In this way, the actuator can turn into both directions, depending on the amplifier input combinations. Table 4.1 summarizes the amplifier states and the corresponding

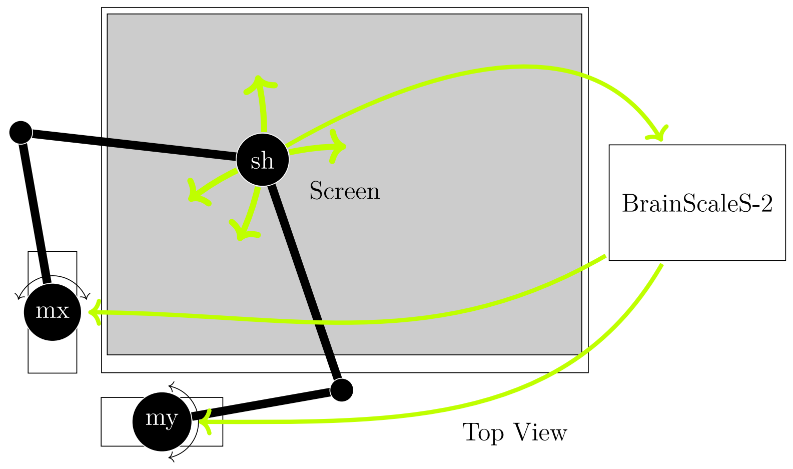


Figure 4.9: The PlayPen. Two actuators (mx and my) pantographically move an optical sensor head (s) over a surface with some graphical content (screen). The sensor information is received and processed by BrainScaleS-2. Motor signals then drive the two voice coil actuators, thus, creating a cybernetic loop between the sensor, the neuromorphic hardware and the actuators.

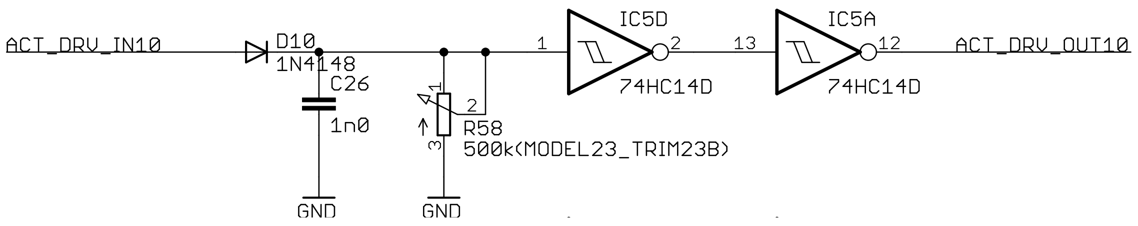
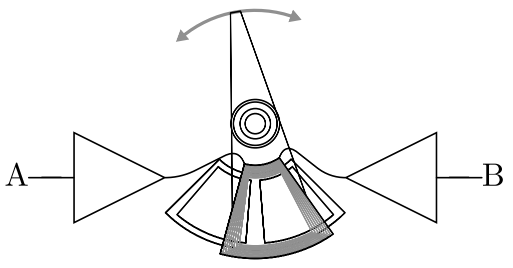


Figure 4.10: Pulse shaper of the PlayPen prototype. `ACT_DRV_IN` denotes the pulse input and `ACT_DRV_OUT` the pulse output. The full schematic can be found in appendix C.4.



A	B	voice coil response
`LOW`	`LOW`	–
`LOW`	`HIGH`	↺
`HIGH`	`LOW`	↻
`HIGH`	`HIGH`	–

Table 4.1: Amplifier output states and voice coil responses.

voice coil responses.

Since the amplifiers remain in low impedance, they source current, in case their output state is `HIGH` and the sink current in case their output state is `LOW`. When both amplifiers are in the same output state, the actuator is not actively driven, as no active current flows through the coil. Because the amplifier outputs remain in low impedance, however, the coil's terminals are virtually shorted, acting as an induction brake. Since this brake acts only when the driving signals are ambiguous (`HIGH HIGH`) or off, it does not negatively impact the motor unit's performance, while motor actions are executed. Because of the low-friction bearings of the voice coil actuators, the induction brake even benefits smoother movements as it slightly damps overshooting and oscillations when the actuator bumps into its mechanical limits.

Taken together, a robotic motor units translates an input spike train into a force response that is proportional to its rate. Given a preprocessed pulse length of 8 µs, a maximum spike rate of 100 kHz causes a duty cycle of 80 % in the amplifier outputs which leaves a margin of 20 % to the maximum response (i.e., always on).

4.3.2 Sensors

As important as the robotic motor units are of course the sensor inputs and the corresponding signal chains. Again, a few basic estimations are made to narrow down the hardware demands.

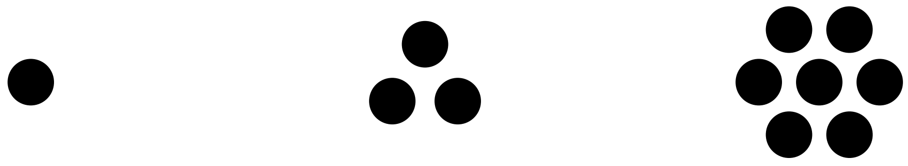


Figure 4.11: Top view of three schematic sensor arrangements with one, three, and seven channels, respectively. Every dot indicates the receptive field of a sensor channel.

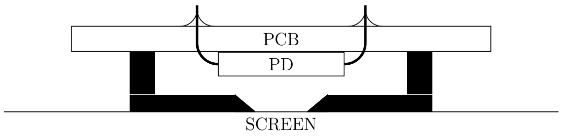


Figure 4.12: Photo diode (PD) within its aperture cabinet above a screen.

Sensor head

As figure 4.9 indicates, the PlayPen's sensor head is supposed to capture visual input that is being displayed on a screen that it is moving on. As explained in section 4.2.2, the restricted number of neurons in the BrainScaleS-2 prototype chips makes it futile to connect high-dimensional sensor input to the neuromorphic chip. However, a large number of experiments can already be facilitated using only low-dimensional sensors.

figure 4.11 illustrates three applicable sensor topologies with one, three, and seven sensor channels, respectively. While the one-dimensional sensor on the left is blind to spatial gradients, the three-dimensional sensor in the middle can resolve linear gradients in 2D. The sensor with seven channels on the right can even resolve 2nd order gradients and local maxima. Within this thesis the one- and three-dimensional sensor types have been evaluated.

Figure 4.12 shows a cross section of one optical sensor. A photo diode (PD) is mounted on the bottom side of a PCB and shielded from ambient light by a plastic aperture that slides directly on top of the screen. In this way, the sensor is spatially selective to the small aperture opening, despite the larger active area of the photo diode. Figure 4.13 depicts a sketch of how the aperture opening is similar to a receptive field applied to the screen content.

To derive an estimate for the photo diode response, it is first assumed that the aperture opening is homogeneous in terms of its effect on the diode. By that is meant, that a point-like light source with a fixed luminous intensity would cause the same electrical effect in the photo diode independent of its position within the aperture opening. If, in addition, the opening is circular with a diameter d_F, then the corresponding kernel function can be expressed as

$$F(x,y) = \begin{cases} \frac{4}{\pi \cdot d_F}, & \text{for } \sqrt{x^2+y^2} \leq \frac{d_F}{2} \\ 0, & \text{else} \end{cases} \tag{4.9}$$

In this case, the diode current only depends on the convolution L of this receptive field F and the screen content S at the position (x, y):

$$L = (F * S)(x,y) \tag{4.10}$$

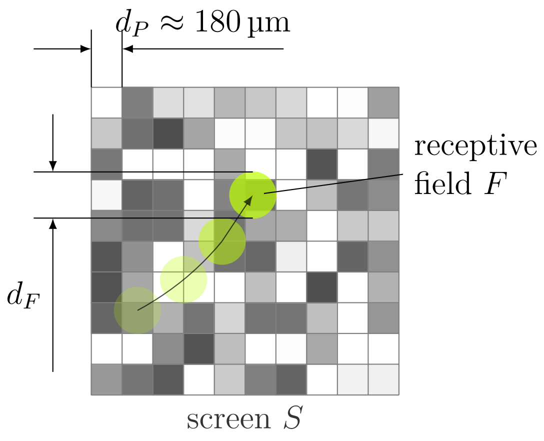


Figure 4.13: Convolution of the receptive field F and the screen content S. The pixel pitch on the screen is roughly $d_P \approx 180\,\mu\text{m}$. The diameter of the receptive field d_F is a little larger than that.

If $d_F \gtrsim d_P$, i.e. the opening covers more than one pixel, F acts like a spatial low pass filter, effectively averaging over the area defined by the aperture. Choosing $d_F \approx d_P$ allows the sensor to access the full screen resolution but potentially demands a higher photo diode sensitivity as more light is blocked by the aperture. Selecting $d_F < d_P$ does not provide any improvements in resolution but only reduces the light and, thus, the current through the photo diode.

When the sensor head is moving, the maximum signal frequency, i.e., the rate at which pixels pass by, is $f_\text{signal} = \frac{v_S}{d_P}$, where v_S is the speed of the senor head. Given the acceleration that was estimated in section 4.3.1, i.e. $a = \mathcal{O}(3000\,\text{m}\,\text{s}^{-2})$, the sensor head speed after accelerating over $s = 1\,\text{cm}$ is $v_S = \sqrt{2as} \approx 7.7\,\text{m}\,\text{s}^{-1}$. Thus, the maximal signal frequency should be on the order of $f_\text{signal} \approx 43\,\text{kHz}$ which corresponds to 43 Hz in biological time. Given that most healthy humans can hardly resolve flicker frequencies that are higher than 40 Hz (Unios Australia Pty Ltd, 2019), this rate is acceptable.

As we aim at maximizing the signal frequency for our accelerated neuromorphic agent, also d_F is chosen to be on the order of d_P to not further reduce the signal rate by spatial filtering.

However, another aspect that has to be kept in mind is the sensitivity of the photo diodes. For their small and lightweight form factor and their good availability, `BPW34` photo diodes (VIS, 2011) were selected. With bias voltage applied, the current through this diode is

$$I_D \approx 11 \cdot P^{1.08}\,[\text{A}] \tag{4.11}$$

where P is the integrated light intensity over the photo-active area in watts. A typical liquid crystal display (LCD) emits a power of approximately 2.4 µW per pixel at full brightness [2]. Assuming a contrast ratio of 1:1000, the light power received by the photo diode is therefore roughly on the order of 2.4 nW to 2.4 µW, which translates to a diode current of 5.4 nA to 9.4 µA. Signals in the nA-range usually

[2]A typical LCD monitor achieves maximum brightness at around 5 W backlight power (Issa, 2012). Assuming that that the light can be efficiently transported through the LCD panel, the approximate power per pixel for a monitor with 1920 × 1080 pixels is therefore $P_\text{pixel} \approx 5\,\text{W}/(1920 \cdot 1080) = 2.4\,\mu\text{W}$.

demand carefully shielded routing in PCB layouts when they are supposed to be transmitted over distances of a few cm. As the signal has to travel from the quickly moving sensor head to the processing circuits, only a fully shielded cable could guarantee the signal integrity. However, as the signal environment is highly polluted by the strong magnetic fields of the voice coil actuators and broad band noise coming mainly from switching regulators and digital circuits on the signal processing board and BrainScaleS-2, even good shielding would not suffice to reliably resolve a few nA over that distance. The diode current is therefore pre-amplified and converted into a voltage directly on the sensor head such that the signal can be transmitted with a higher signal-to-noise ratio to the signal processing board. This amplification is optimally set such that the pre-amplified signal amplitude spans the accessible input voltage of the receiving signal processing units. The remaining part of the signal chain that converts the signal into a spike train is implementation-specific to the robot version. Therefore, details are given in the corresponding following sections.

However, if even the preamplifier is not sufficient to reliably amplify 5.4 nA, the aperture diameter d_F can be increased to allow more light to hit the photo diode surface. Note that, to our advantage, the decrease in signal frequency scales linearly with d_F, while the integrated light intensity scales quadratically with d_F.

Moreover, the experiments that were performed so far, did not rely on a full dynamic range of 1:1000. As long as the maximum spike rate corresponds to the maximum light intensity, structure in the visual input can still be resolved by the sensor, loosing only a certain amount of precision in worst case.

Rotary position

A second category of sensors are rotary encoders that measure the position of the voice coil actuators. Much of the related part selection and evaluation has been carried out in Louis Jussios' project internship (Jussios, 2018). The relevant selection criteria for the sensor type are its mechanical properties, the sampling rate, and the precision. As the friction and momentum of the actuator are to be kept possibly low in order to not unnecessarily decrease its speed and acceleration, some form of contactless measurement is ideal. During the design processes, especially optical and magnetic sensors have been considered. The initial choice fell onto optical Gray code (Frank, 1953) sensors, as they only require a lightweight printed add-on on the actuators and a relatively simple arrangement of photo sensors and illumination that is mounted in a static position. It became clear, however, that a reasonable level of precision was technically hard to achieve because the sensor resolution scales with the size of the add-on that has to be attached to the actuator. Moreover, the entire arrangement would have required a non-negligible amount of engineering and testing which would have gone somewhat too far beyond the intended focus of the PlayPen project. Gray encoders were therefore discarded and replaced by magnetic Hall-effect-based sensors. Jussios selected the `AS5600` programmable contactless potentiometer (ams, 2018) for its high angular resolution and sampling rate, its simplicity, and its good availability.

The only component that contributes additional weight to the moving actuator is a small neodymium magnet that has to be mounted on top of the center of rotation[3].

Together with a custom 3D-printed mounting socket, one magnet weighs about 1 g. As its radius is on the order of a few mm and as it is mounted at the very center of rotation, it does however only contribute negligibly to the total moment of inertia of all moving units.

The `AS5600` has a measurement resolution of 12 bit that can be mapped to either the full range of 360° or to a programmable arbitrary subrange. As it takes one sample every 150 µs, its sampling rate is approximately 6.67 kHz, corresponding to 6.67 Hz in the biological domain. Although a higher sampling rate would be desirable in order to better match the estimated lower limits of biological sensor response times (see section 4.2.2), an interval of 150 ms biological time is still close to the average spike time intervals and therefore acceptable. Moreover, the reaction time of the system is ultimately limited by the mechanical restrictions of the actuators.

The `AS5600` can be configured and read out via a digital I^2C interface at a maximum data clock speed of 1 MHz. For each data readout, a slave address and a request address have to be transmitted to the sensor IC first. Then the chip transmits the measured 12 bit position packed in two bytes. Thus, the achievable sample data rate per sensor is 31.25 kHz. A drawback of the `AS5600` is that it has a static, non-configurable I^2C slave address. As a result, only one `AS5600` can be connected to one I^2C-bus since otherwise all `AS5600` would respond simultaneously to a read/write request. Since two sensors are required to access both actuator positions, this problem has to be circumvented. Instead of utilizing a second I^2C bus channel, the I^2C clock signal (which is always driven by the I^2C master) is gated to one or the other `AS5600` by a custom logic circuit. Thus, in order to access a specific `AS5600`, the I^2C master has to select the respective clock gate and initialize the normal data request procedure. Hence, the sensors can only be read out alternatingly, effectively halving the digital readout rate to 15 625 Hz. As this rate is still higher than the internal data sampling rate, no disadvantage is caused by this procedure.

In addition to the digital output, the `AS5600` contains an internal DAC that can stream out an analog voltage representing the rotary position. Like the digital readout, also the analog output can be configured such that only a selected angular region is mapped onto the entire output voltage range.

In effect, the rotary position can be represented in the same format as the optical sensors, i.e., such that it covers the accessible input voltage of the receiving signal processing units. The signal processing is again experiment- and version-specific and is therefore explained in detail in the corresponding next sections.

4.3.3 Prototype

The first incarnation of the PlayPen is to be seen as a proof-of-concept prototype system that lacks a large number of possible optimizations. Figure 4.14 gives a schematic overview of the system components. In the following, subsections which concern electrical modules are sometimes accompanied by thumbnail pictures of the

[3]Figure 4.28 shows this magnet attachment for the second PlayPen version.

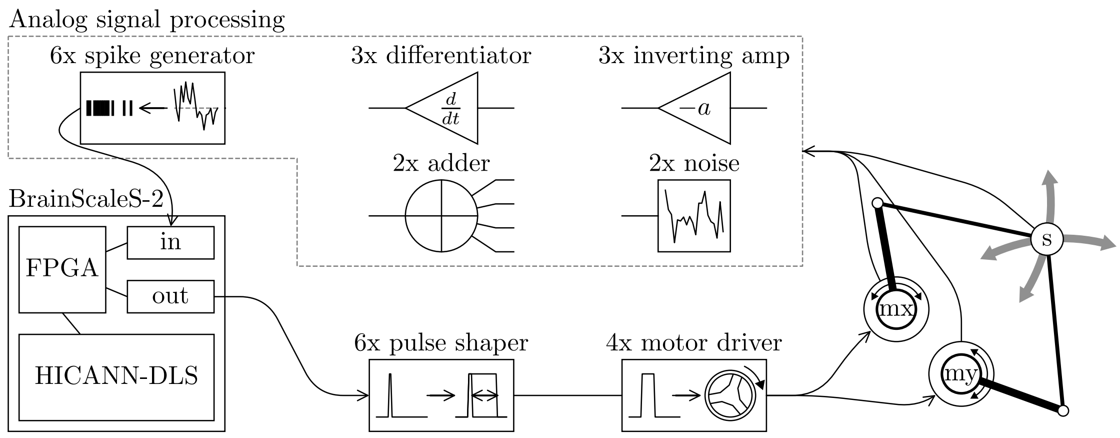
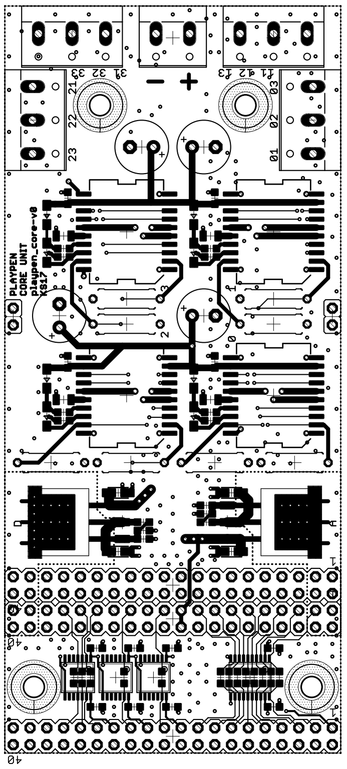


Figure 4.14: PlayPen prototype. The digital spike output from BrainScaleS-2 can be received by up to six pulse shapers (see figure 4.10 and section 4.3.1) that deliver a preprocessed pulses to up to four motor drivers, of which each is responsible for one actuator direction. The pre-amplified analog signals from the sensor head and the rotary encoders are sent to an analog signal processing unit, which contains three inverting amps, three differentiators, two quad-input adders, two white noise sources and six spike generators. The latter can be connected back to the BrainScaleS-2 spike input.

corresponding PCBs to give the abstract descriptions a little more character. If no explicit reference is given, these images always relate to the respective sections.

Core module



The power supply for all components except BrainScaleS-2 is implemented together with the motor driving circuits and the `TXB0104/8` level shifters on a PCB referred to as *core module*. The entire schematic can be found in appendix C.4. This module hosts only components that are universal to all PlayPen experiments and does not contain parts of the actual signal processing. Instead, it has a parallel interface offering BrainScaleS-2 spike outputs at preprocessed voltage levels (5.0 V), supply voltages (5.0 V digital and analog), and motor driver inputs to which a daughter board can be connected. During the lifetime of the PlayPen prototype only one daughter board, the *HC14 module*, has been commissioned and used. It is described in detail further below and its schematic can be found in appendix C.5.

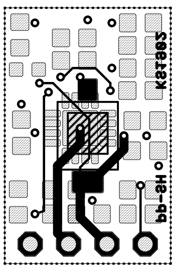
During the initial planning phase of the PlayPen project, a few different actuator types were considered, which is why the core module contains up to four `L6234PD` three-phase motor drivers (STM, 2017) of which only two are equipped and necessary to drive the two HDD voice coils. The `L6234PD` can deliver up to a 5 A of pulsed output current at a commutation frequency of up to 150 kHz. They are supplied with a voltage of 12 V and therefore capable of delivering 60 W pulses into the coils, satisfying the expected demands of equation (4.3).

Since the voice coils and magnets are precisely mounted within the sophisticated HDD aluminum cabinets, the easiest way to preserve their intended position with respect to each other is to leave them in their cabinets. Therefore, everything but the magnets and actuators has been removed from the HDDs priorly and the cabinets were trimmed to the minimum necessary size with a band saw. The disadvantage is that the voice coil tips then partially overlap with the remaining aluminum frames, blocking the mechanical access to them. In the prototype version, this problem is solved by spatially extending the actuators using rigid and lightweight carbon fiber tubes. Each actuator is covered by a piece of cardboard that holds an `AS5600` evaluation board (ams, 2014) in a suitable position above the moving magnet.

The actuator movements are mediated to the sensor head by two joints and another pair of carbon fiber tubes. One of these tubes is rigidly connected to the sensor head, while the other one is flexibly mounted by another custom joint.

Both, the sensor head and the rotary sensors are supplied by 5.0 V coming from the analog signal processing board.

Sensor head



The sensor head consists of a custom PCB and a 3D-printed cabinet that shields the sensor from ambient light while guiding the relevant light to its center. The photo diode sits on the bottom side, the preamplification circuits on the top side of the PCB. The circuit is centered around a `MAX4206` logarithmic transimpedance amplifier (max, 2015), that is particularly designed for photo diode applications. The ideal transfer function of this amplifier is

$$V_{\text{out}} = K \cdot \log_{10} \frac{I_D}{I_{\text{ref}}} \tag{4.12}$$

where K as well as I_{ref} are configurable by connected passive components. Since the output voltage range of the amplifier is specified as 0.08 V to 4.8 V, and since the analog processing board can handle rail-to-rail voltage input [4], K and I_{ref} are configured such that the expected diode current of 5.4 nA to 9.4 µA is mapped to the full range. Being a logarithmic amplifier, the `MAX4206` also converts the logarithmically distributed brightness of the screen into linearly distributed voltages, which is of great advantage for the subsequent signal processing on the HC14 module.

[4]Rail-to-rail means that the allowed or achievable voltage covers the entire range from the negative supply voltage to the positive supply voltage of the corresponding electronic component. E.g., for a component that is supplied by $V_{\text{DD}} = 5\,\text{V}$ and $V_{\text{EE}} = 0\,\text{V}$, rail-to-rail input means that all voltages within 0 V to 5 V are allowed and can be processed.

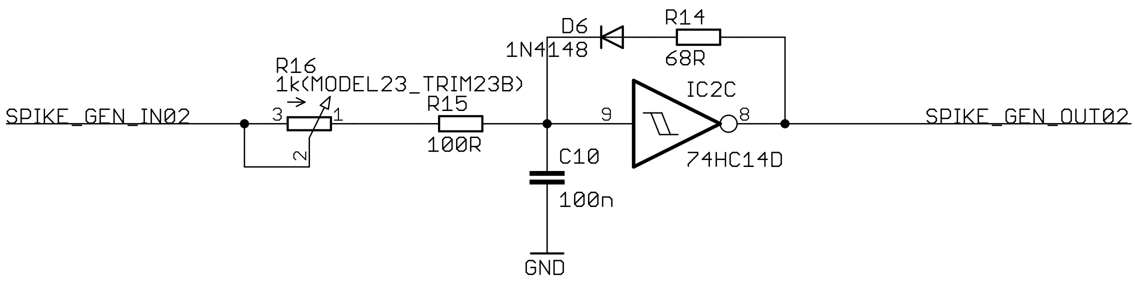
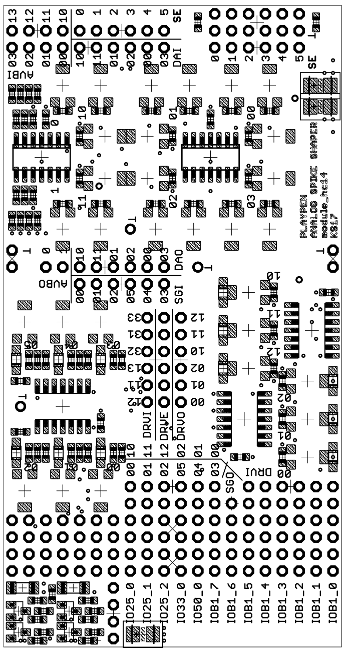


Figure 4.15: Voltage to spike converters of the PlayPen prototype.

HC14 module



In order to facilitate a high level of flexibility, the processing module contains a variety of analog subcircuits that can be patched together as needed. The name *HC14 module* refers to the extensive use of the `74HC14` six-channel Schmitt inverters (Nex, 2020). Section 4.3.1 already described how a Schmitt inverter can be employed for extending the width a digital pulse. Another application is to convert an analog voltage into a pulse train with a frequency that is proportional to this voltage.

Figure 4.15 shows the schematic of such a unit. To understand its operation, let us first assume an input voltage $V_{\mathrm{in}} = 5\,\mathrm{V}$ that charges the capacitor C over the potentiometer R_P and a resistor R_R. When a stationary state is reached, the voltage at the input of the Schmitt inverter, V_{s}, is $5\,\mathrm{V}$ as well and its output voltage is $0\,\mathrm{V}$. As no significant current flows through the diode D when a reverse bias is applied, the stationary state is maintained as long as V_{in} does not change. When the V_{in} decreases, V_{s} follows until it reaches V_{TN}, which causes the Schmitt inverter to change its output voltage to $5\,\mathrm{V}$ and, in turn, charge the capacitor C over the diode D and the feedback resistor R_F increasing V_{s} again. Since R_F is much smaller than $R_P + R_R$, the current through the feedback path dominates the charging during this phase. When V_{s} reaches V_{TP}, the output state changes back to $0\,\mathrm{V}$ and C discharges again via R_P and R_R. The cycle continues as long as $V_{\mathrm{in}} < V_{\mathrm{TN}}$, causing a regular train of pulses. The pulse length is dominantly determined by the resistance of D, R_F, and the capacitance C. The pulse frequency on the other hand depends mostly on V_{in}, R_R, R_P, and C.

Hence, for $V_{\mathrm{in}} < V_{\mathrm{TN}}$, the spike rate is proportional to $-V_{\mathrm{in}}$. The sensitivity can be tuned by the potentiometer R_P. This circuit is implemented six times on the HC14 module and represented as the *spike generators* in figure 4.14.

Another set of units on the processing board are the inverting amplifiers. These are operational amplifier-based circuits that each have an independently configurable offset and amplification. Two potentiometers are used for each of the three amplifiers to set these parameters.

To give an application example, consider an inverting amplifier that maps an

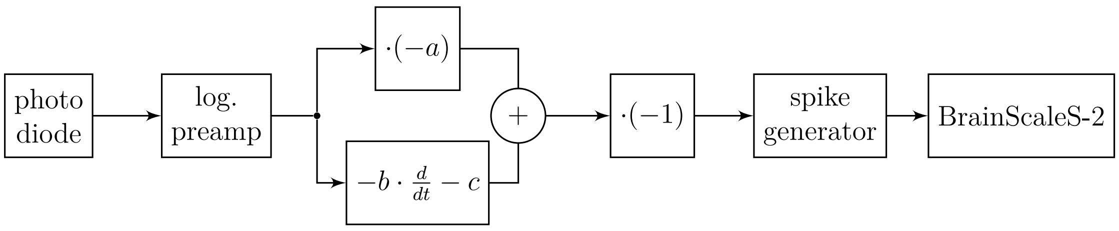


Figure 4.16: Block schematic of a signal chain that implements equation (4.4) to mimic sensor unit spike responses as depicted in figure 4.6.

incoming sensor voltage to the operating range of such a circuit. A spike train can then be generated whose rate is proportional to the sensor input. In that way, simple rate encoding can be implemented.

To achieve response functions like in figure 4.6 the HC14 module further contains a set of three inverting differentiators, again with potentiometer-defined amplification and offset for each unit.

Linearly amplified and shifted sensor signals can be combined with scaled and shifted signal derivatives using the two quad-input adders. A superposition of terms like in equation (4.4) can thus be realized. A block schematic of this arrangement is given in figure 4.16.

The last subcircuits to be discussed are the two white noise sources. The intension behind these is to introduce an easily accessible source of stochasticity into the module that can be used to create a random spike train together with the spike generators, or to smooth out artifacts that might occur due to strong nonlinearities. For example, the transition from $V_{\text{in}} > V_{\text{TN}}$ to $V_{\text{in}} < V_{\text{TN}}$ causes a sudden increase in the output rate of the spike generators. If noise is added to V_{in}, the transition becomes stochastic and therefore smoothes out similar to figure 2.9.

The complete schematic of the HC14 module with all its subcircuits can be found in appendix C.5.

The only component that has been added retrospectively to the initial prototype setup is an `Arduino Nano` (Ard, 2008) with the above mentioned I^{2}C clock gating circuit to digitally access and configure the `AS5600` rotary sensors. The `Arduino Nano` is a low cost and easy to use microcontroller platform with integrated power supply and USB interface. It is based on an 8-bit `ATmega328P` microcontroller that runs at a clock frequency of 16 MHz. Due to a user-friendly and comprehensive open-source integrated development environment (IDE), code for the `Arduino Nano` can be developed quickly and without deep prior knowledge on microcontrollers, which makes it perfect for prototyping small research projects, like the PlayPen. The power supply for the I^{2}C clock multiplexing is also provided by the `Arduino Nano`, such that the entire device does not require more than a USB-connection to the host PC. Further details on the implementation and usage can be found in Jussios (2018).

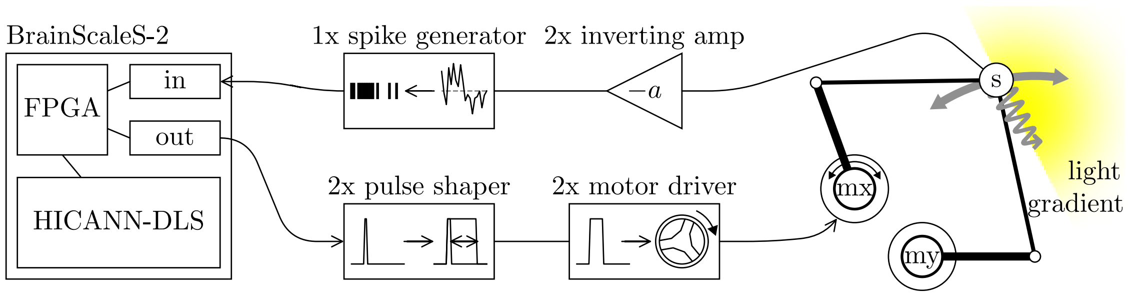


Figure 4.17: Experiment schematic. The sensor head is supposed to trace the edge of a light gradient. Only one arm (mx) is enabled, the other one is left loose.

4.4 Edge tracing

To test the entire signal chain of the PlayPen prototype, a simple neural network was implemented that uses only one motor unit to trace a border between light and shadow. The experiment was presented during the 12th CapoCaccia Cognitive Neuromorphic Engineering Workshop. Being more of a proof-of-concept demonstration, the results are displayed only qualitatively without extensive analysis.

Figures 4.17 and 4.18 show the block schematics of the signal chain and the neural network. The HC14 module is used such that the scalar brightness, as measured by the sensor head, is turned into a spike rate. This spike train then enters BrainScaleS-2 via its spike input where it is received by two neurons (N0 and N2) via one synaptic circuit, each. Both neurons only act as repeaters of the incoming signal. N1 on the other hand is neuron that is configured to spike without external excitation. This is achieved by setting its resting potential higher than the spiking threshold $V_{\mathrm{leak}} > V_{\mathrm{thresh}}$. Its reset potential V_{reset} is set to be smaller than V_{thresh}, such that its membrane potential V_{mem} has to recover after each spike before a new spike is triggered. The time it takes for V_{mem} to increase from V_{reset} to V_{thresh} is proportional to the difference between V_{thresh} and V_{reset}, the membrane time constant τ_{mem}, plus the refractory period τ_{refr}. These parameters, therefore, allow setting the resting spike rate to a moderate level that is equal to the rate of N0 at maximum sensor input.

The role of N2 is to reduce N1's firing rate in case of sensor input. To do so, it is connected presynaptically to an inhibitory synapse of N1.

Finally, N0 is connected to the motor unit that moves the actuator into clockwise direction, while N1 is connected to the one that drives it counterclockwise,

The network can be thought of as a buffer (N0 branch) and an inverter (N2-N1 branch) that receive the same sensor input. If the input is weak, the voice coil will rotate counterclockwise, if it is strong, it will turn clockwise. If the sensor input is such that a clockwise rotation weakens the light, while a counterclockwise rotation causes the sensor to move to a brighter region, the system will act as a regulator that keeps the sensor head at a position of constant brightness (see figure 4.17).

To create the light gradient, half of a light cone that came from below the sensor head was coverd with a sheet of paper. After setting up the chip, the network started to trace the paper edge.

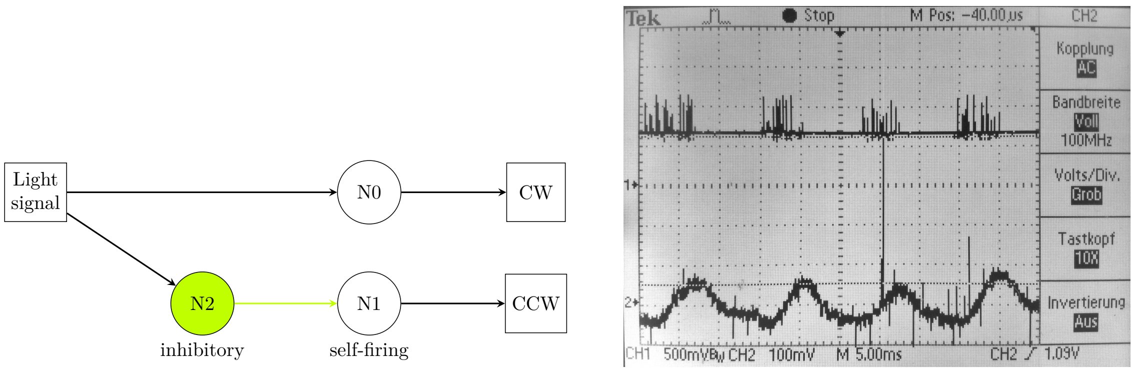


Figure 4.18: Left: Block schematic of the edge tracing experiment. The signal coming from the light sensor is first turned into a spike rate using the inverting amplifiers and a spike generator. After entering the BrainScaleS-2 Spike I/O, the spike train excites two neurons, N0 and N2. N2 then inhibits a third neuron N1 that is self-firing, that is, firing without external excitation. N0 is connected to the amplifier that makes the actuator move clockwise and N1 is connected to the amplifier that turns it counterclockwise. Right: Oscilloscope traces of the rotary position (bottom trace) and the spike response (top trace) over time.

Figure 4.18 (right) shows two signals, the analog voltage obtained from the rotary position sensor and the spike output of N1. The latter is not depicted in full resolution because the rate is on a much smaller time scale than the movement. Well visible is however that an oscillation period is on the order of 13 ms. Thus, the reaction time of the actuator is too slow to correspond to a biological muscle system, like a limb for example (see section 4.2.1). However, the experiment successfully closes the loop between an accelerated neuromorphic system and a mechanical robotic actuator.

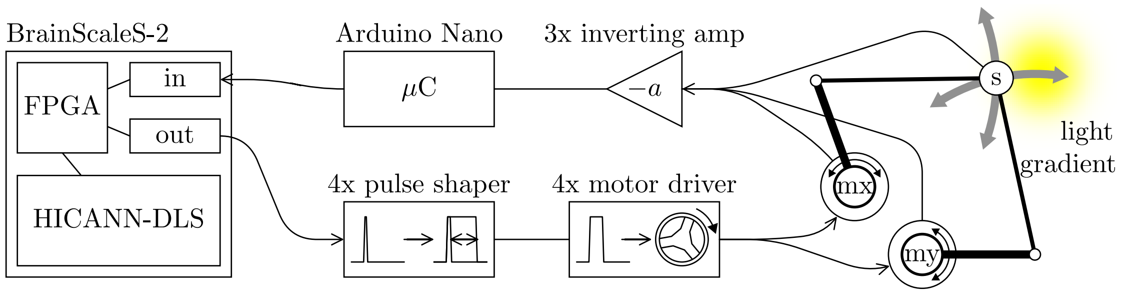


Figure 4.19: Experimental setup for a neuromorphic maximum finder with a scalar light sensor signal.

rotation x	rotation y	movement
–	ccw ↺	↑
cw ↻	ccw ↺	↗
cw ↻	–	→
cw ↻	cw ↻	↘
–	cw ↻	↓
ccw ↺	cw ↻	↙
ccw ↺	–	←
ccw ↺	ccw ↺	↖

Table 4.2: Translation of arm rotations into directions of sensor head movements. Counterclockwise rotations are symbolized as ccw ↺, clockwise rotations by cw ↻. The resulting movement direction of the sensor head is indicated by arrows whose directions are in correspondence with figures 4.9, 4.14 and 4.19. Adapted from Prinzler (2018).

4.5 Maximum finder on prototype

The goal for this experiment was to track and trace the location of maximum brightness in a light field that comes from below the sensor head.

Contribution

The following experiment has been implemented in the course of Malte Prinzler's Bachelor's theses (Prinzler, 2018). Malte Prinzler and the author of this thesis both contributed to the experiment design and execution. Measurements and plots where created by Malte Prinzler.

As the PlayPen prototype only contains a one-dimensional optical sensor, the light gradient cannot be measured spatially but only be deduced by correlating movement with temporal changes in the sensor signals. A block schematic of the experimental arrangement is given in figure 4.19. To get information on the sensor head motion in two dimensions, the derivatives of both actuators are required. Table 4.2 provides a summary of how the actuator rotations correspond to the directions of sensor head movement.

Initially, the signal derivatives where constructed by the differentiators on the HC14 module (see section 4.3.3). However, a problem that became apparent very soon, was that the high frequency noise on the incoming analog signals was massively dominating the differentiator response. To circumvent this, a series of passive analog low pass filters, preceding the differentiators, was inserted into the signal path. While functional differentiation could be achieved with this extended analog network, the processed signal was still very noisy and tuning became increasingly complicated as all circuit components had to be matched well to the speed and response time of the system, which constantly changed during experiment development. Eventually, the decision was made to replace the analog differentiators with better controllable and configurable digital filters. In order to keep the implementation low in effort, easy, and flexible, the signal processing has been implemented on another `Arduino Nano` (Ard, 2008). The signals were first amplified and shifted by three inverting amplifiers on the HC14 module and then digitized by the `ATmega328P`'s internal ADCs on the `Arduino Nano` board. These ADCs have an input range of $0\,\mathrm{V}$ to $5\,\mathrm{V}$, a resolution of $10\,\mathrm{bit}$, and a maximum sampling frequency of $10\,\mathrm{kHz}$. Since the `Arduino Nano` contains no analog output but instead a large amount of digital outputs, the spike generation has been implemented digitally as well. After optimizing the code, Prinzler achieved a maximum output rate of $125\,\mathrm{kHz}$ per channel at a sampling rate of $1.778\,\mathrm{kHz}$. While the maximum spike rate is optimal, the sampling rate only corresponds to about one sample every 0.56 seconds. Considering the relatively low mechanical agility of the prototype system and the simplicity of the input data (a smooth light gradient), the sampling rate does represent the limiting factor in terms of dynamical performance however. The temporal changes in the sensory input are expected to be on the same order as the sensor head motion and therefore on the order of maximally $390\,\mathrm{Hz}$ (see equation (4.8)). As the sampling rate is therefore higher than the Nyquist frequency (Nyquist, 1928) $f_{\mathrm{sample}} > 2 \cdot f_{\mathrm{signal}} = f_{\mathrm{Nyquist}}$, it is considered sufficient.

The first digital filter that is applied to the signal $x(t)$ after being digitized is an exponentially smoothing moving average. This filter is computationally efficient and requires only one word (in this case two byte) of memory allocation per channel. The moving average is

$$\langle x(t) \rangle = \kappa \cdot x(t) + (1 - \kappa) \cdot \langle x(t-1) \rangle \tag{4.13}$$

with the discrete time t and the smoothing coefficient $\kappa \in [0; 1]$.

Since the `ATmega328P` does not contain a floating point unit and can therefore only implement integer operations efficiently, κ is expressed as a division of two integers $\kappa = \alpha/\beta$, where the denominator $\beta = 64$ is fixed and α is selectable between 1 and 64. Extensive details on the filter response function in the time as well as in the frequency domain can be found in Prinzler (2018). This moving averaging stage is used to reduce input noise by filtering out high frequency components that do not represent the actual sensor inputs but are inevitably introduced throughout the analog signal path.

The next stage are the differentiators. On a discretized signal, their output is proportional to the difference between the input at t and at $t - \Delta t$:

$$\Delta \langle x(t) \rangle = \langle x(t) \rangle - \langle x(t - \Delta t) \rangle \tag{4.14}$$

The time difference Δt is, like α, a tunable parameter that is subject to extensive analysis in Prinzler (2018). α and Δt together can be thought of describing the lower and higher cutoff frequency of a band-pass filter on the sensor signals. Note that, $\Delta \langle x(t) \rangle$ is only a proxy for the actual derivative $\dot{x}(t)$. In particular, a constant factor of dimension $1/t$ is missing that scales $\Delta \langle x(t) \rangle$ in amplitude and accounts for the variable time interval Δt. However, since the scaling of the output spiking rate with $\Delta \langle x(t) \rangle$ is arbitrary and internal to the microcontroller, this factor is absorbed in the spike conversion anyhow.

Ultimately, three derivatives are obtained that represent an increase in light intensity and counterclockwise rotations of the two actuators, respectively. By inverting the first one, an additional signal is derived that represents a decrease in light intensity, leaving us with four signals in total, each of which is represented as a spike rate:

- channel 1: increasing light
- channel 2: decreasing light
- channel 3: counterclockwise rotation of actuator my
- channel 4: counterclockwise rotation of actuator mx

If channels 3 or 4 are not spiking, my or mx are either rotating clockwise or they are resting.

With these input signals available, the brightness tracing algorithm can be expressed as a finite state machine (see table 4.3).

The state machine takes action on all four actuators, more precisely, on the motor unit inputs that are responsible for exerting clockwise and counterclockwise torque on both actuators mx and my:

- channel 5: exert clockwise torque on my
- channel 6: exert counterclockwise torque on my
- channel 7: exert clockwise torque on mx
- channel 8: exert counterclockwise torque on mx

When the regulation mechanism behaves according to table 4.3, the resulting sensor head motion is circular around the point of maximum brightness, ideally spiraling into the center. The neural network that implements this state machine is depicted in figure 4.20.

The first layer receives the incoming sensor signals and distributes them to the next layers. The second layer, the *integrating neurons*, are special neurons that are configured such that they respond only when at least a certain number of presynaptic neurons are spiking. The respective number of simultaneously spiking presynaptic partners that are necessary to invoke a response from one such neuron is specified as the number to the lower right of the neuron index. Thus, this layer can be thought

		input					output			
		light	light	my	mx		my	my	mx	mx
motion	light	$\uparrow$	$\downarrow$	$\circlearrowleft$	$\circlearrowleft$	steering	$\circlearrowright$	$\circlearrowleft$	$\circlearrowright$	$\circlearrowleft$
$\leftarrow$	–	0	0	0	0	–	0	0	0	0
$\leftarrow$	$\uparrow$	1	0	0	0	$\leftarrow$	1	0	1	0
$\leftarrow$	$\downarrow$	0	1	0	0	$\uparrow$	1	0	0	1
$\uparrow$	–	0	0	0	1	–	0	0	0	0
$\uparrow$	$\uparrow$	1	0	0	1	$\uparrow$	1	0	0	1
$\uparrow$	$\downarrow$	0	1	0	1	$\rightarrow$	0	1	0	1
$\rightarrow$	–	0	0	1	1	–	0	0	0	0
$\rightarrow$	$\uparrow$	1	0	1	1	$\rightarrow$	0	1	0	1
$\rightarrow$	$\downarrow$	0	1	1	1	$\downarrow$	0	1	1	0
$\downarrow$	–	0	0	1	0	–	0	0	0	0
$\downarrow$	$\uparrow$	1	0	1	0	$\downarrow$	0	1	1	0
$\downarrow$	$\downarrow$	0	1	1	0	$\leftarrow$	1	0	1	0

Table 4.3: State matrix of the maximum finder. $\uparrow$ indicates a signal increase, $\downarrow$ a signal decrease. $\circlearrowleft$ represents a counterclockwise and $\circlearrowright$ a clockwise rotation or torque. If a change in brightness is detected a motor action is executed in accordance with the current direction of sensor head movement. When the brightness does not change significantly, no action is taken by the system. The resulting motion is circular. Adapted from Prinzler (2018).

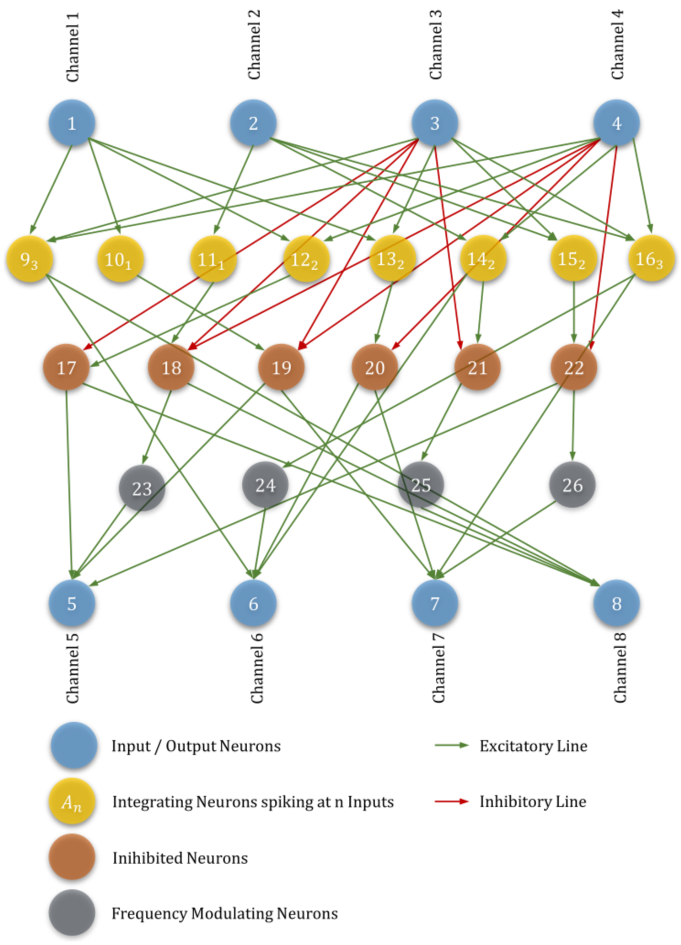


Figure 4.20: Neural network for maximum brightness tracing. The network is organized in five layers, four input neurons that repeat the incoming sensor spike trains, a layer of eight integrating neurons that implement some arithmetic operations, a layer of six inhibited neurons, four frequency modulating neurons, and finally, four output neurons.

of as a layer of parallel neural AND gates[5]. The next layer, the *inhibited neurons*, receive excitatory input from the integrating neurons, as well as inhibitory input from the input layer. Due to the direct connections to the latter, inhibitory input always arrives first. In this way, it is ensured that the neurons in this layer only fire when inhibition does not prevent them from doing so. Transient output spikes that might occur if the spike trains arrived simultaneously is therefore prevented.

The neurons in the fourth processing layer, the *frequency modulating neurons*, have a long refractory period in order to respond with an output rate that is lower than the input rate. Thereby, they translate the high internal spike rates of up to 125 kHz into lower rates of approximately 16 kHz. These neurons are employed to facilitate terraced, gradual output signals. For example, when the brightness increases while the sensor head is actively driven, the motor unit input is supposed to be weaker than in a situation where a correction on the currently taken motor action is required. This allows for quick motor actions when corrections are needed and prevents overshooting due to overproportionate motor actions in situations where the correct actions are already being taken.

Figure 4.21 shows six recorded two-dimensional trajectories of the sensor head during experiment execution. The six recordings have been performed for different parameter settings of Δt (Δn in the plots) and α. The trace brightness of each trace is proportional to the measured brightness at the respective locations. The red dot indicates the point of the maximum measured light intensity, i.e. the expected center of rotation. In addition to the x-y-traces, the distance over time with respect to this point is shown as well in the respective lower plots.

The results show that the maxima are successfully being traced by the neural regulators for a variety of parameter settings. Figure 4.22 shows the neural input and output signals, and the raw sensor readings for one regulation orbit.

Although figure 4.21 illustrates the regulation behavior only for static light locations, the system can also follow moving light sources. Moreover, due to the differential sensor encoding, the absolute brightness of the light source can vary to a fairly high degree.

For example, a flashlight can be held below the sensor head which then interactively follows its position. Figure 4.23 illustrates the regulation for such a scenario. While the brightness remains almost constant during the entire measurement, the slowly changing angular positions reflect the flashlight movement.

The experiment demonstrates that even without a directly accessible spatial gradient, the system is capable of dynamically tracing the maximum in a two-dimensional potential landscape.

[5]An AND gate in Boolean logic is a circuit with n inputs and one output. The output state is 1, only when all n inputs are 1, otherwise it is 0.

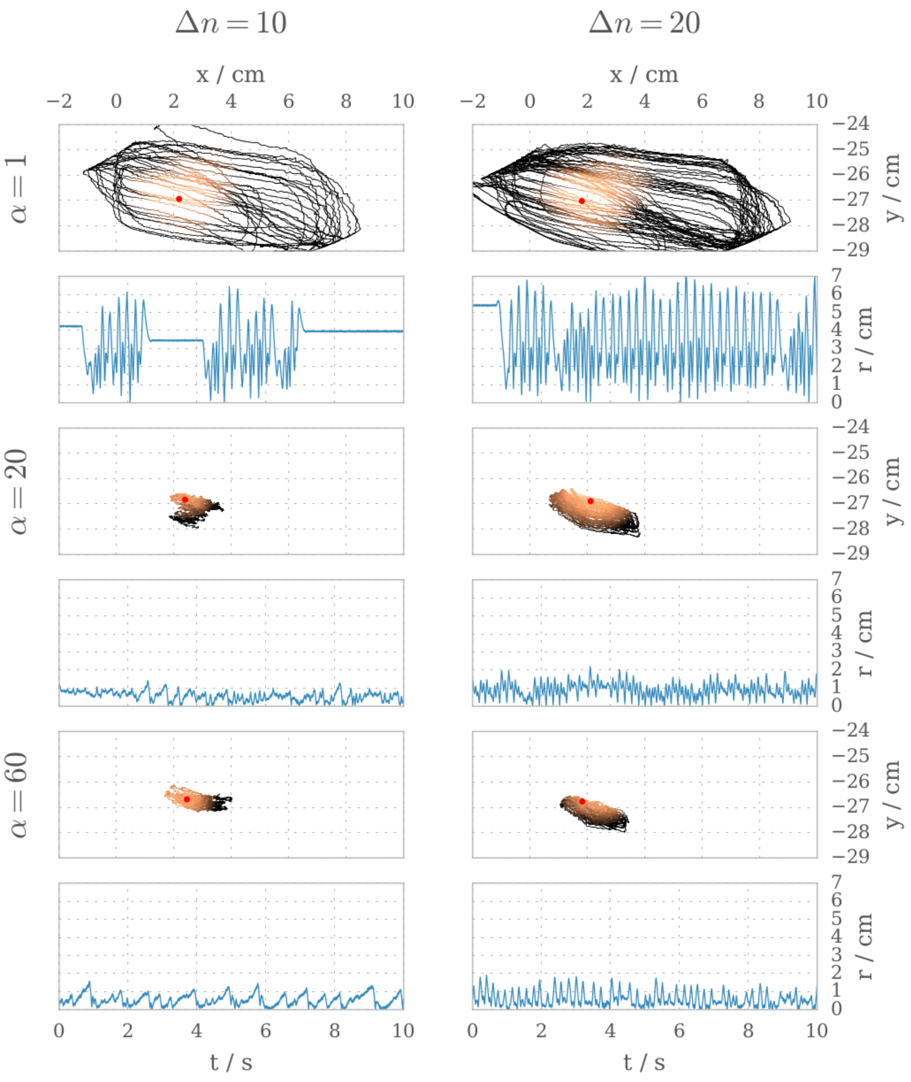


Figure 4.21: Position of the sensor head and distance to the center of maximum brightness during active regulation for different parameter sets. The trace color in the position plots indicates the measured brightness at the corresponding location. The red dots mark the points of the highest measured light intensity. Taken from Prinzler (2018).

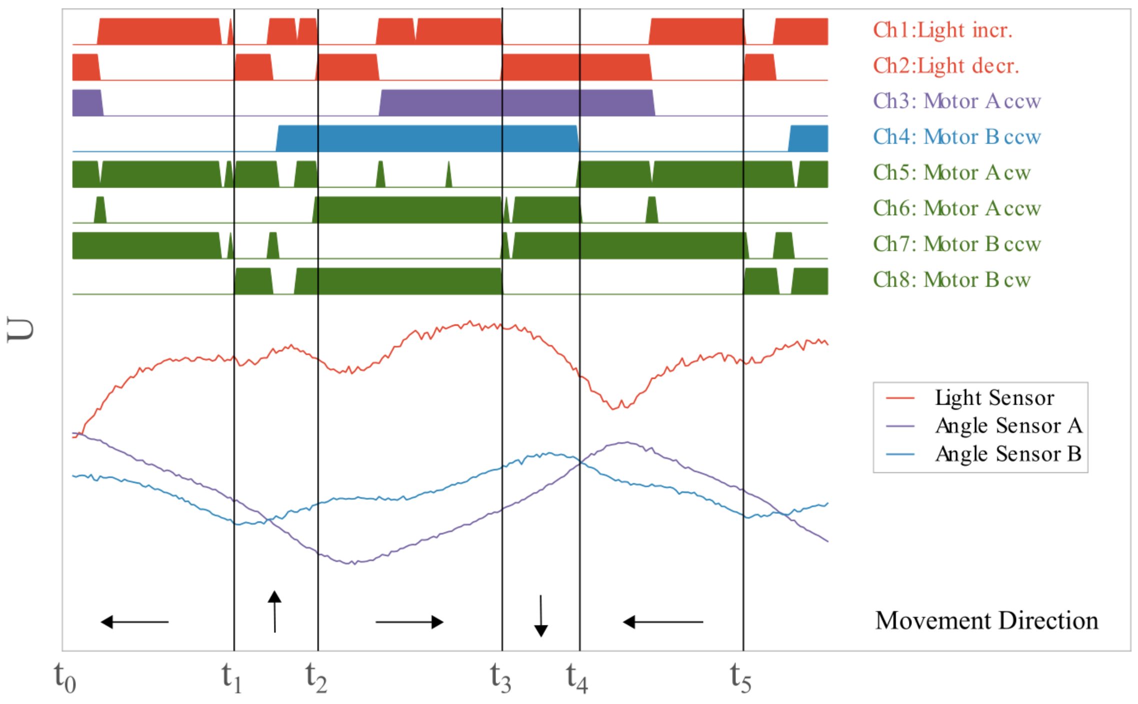


Figure 4.22: Processed and unprocessed sensor input and motor outputs during a regulation cycle. The arrows at the bottom indicate the respective periods in terms of sensor head movement during one orbit. Taken from Prinzler (2018).

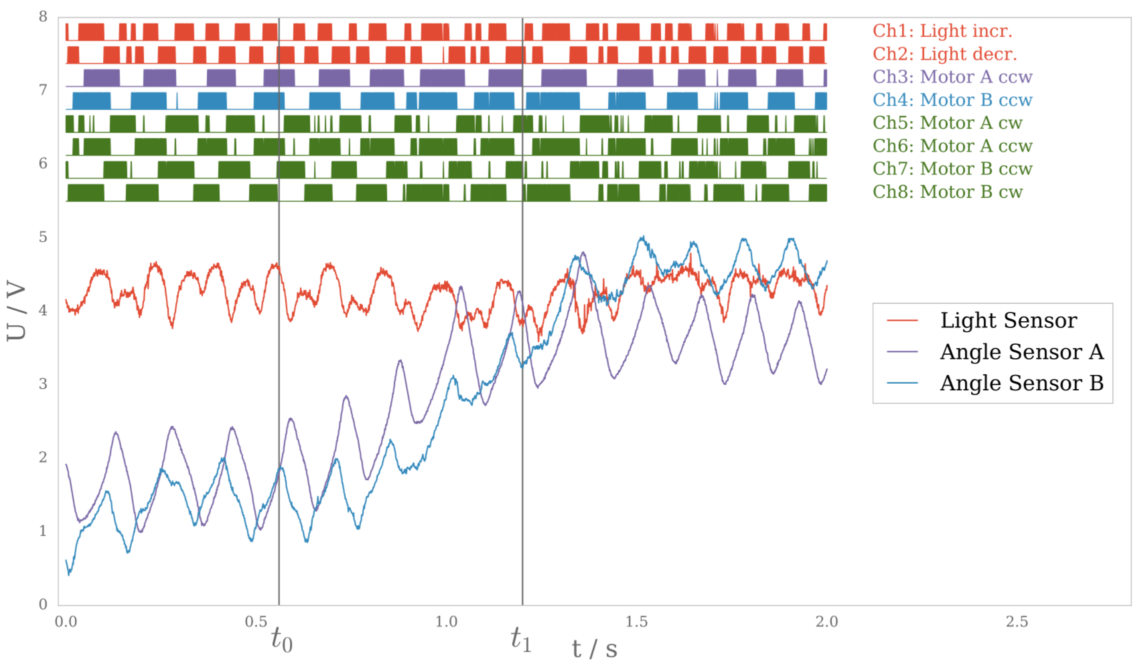
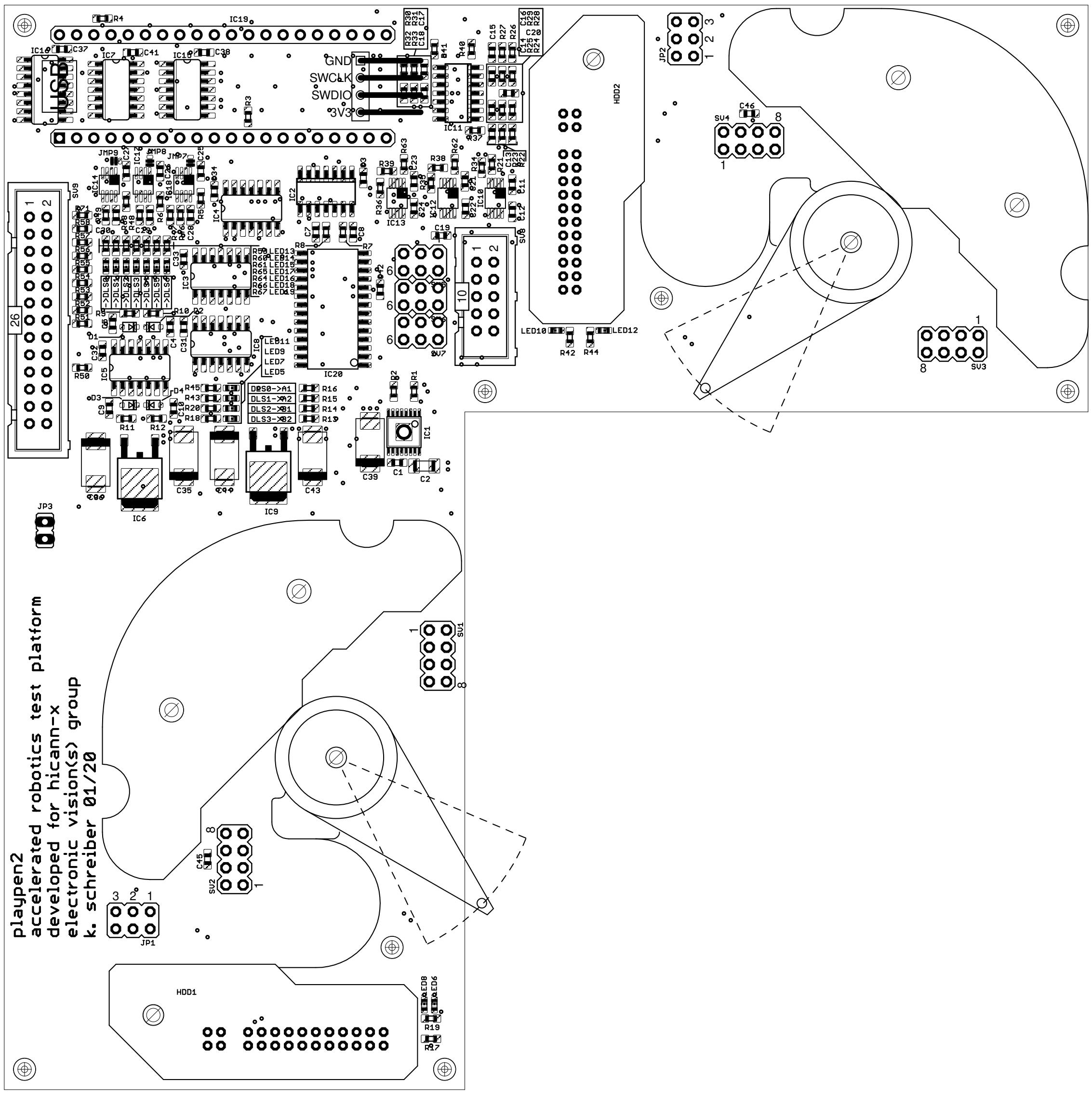


Figure 4.23: Processed and unprocessed sensor input and motor outputs while the location of maximum brightness is moving. While the brightness remains largely constant, the angular positions reflect the light source movement. Taken from Prinzler (2018).

4.6 PlayPen-2



The experiments with the PlayPen prototype revealed a number of bugs, flaws and inconveniences. While some of them only interfere with the convenience of the system, others have a disruptive impact on the functionality. A overview of errors and improvable aspects is given below.

- The bidirectional `TXB0104/8` level shifters on the core module are wrongly connected (VCCA < VCCB, see Tex (2018b,c) and appendix C.4).
- The large number of potentiometers on the HC14 module makes it hard to keep track of the parameter settings.
- Analog differentiation is practically not possible due to noise that is predominately collected by the long signal traces, in particular the patching wires.
- The three-phase motor drivers are obsolete because the actuators were finally implemented as HDD voice coils.

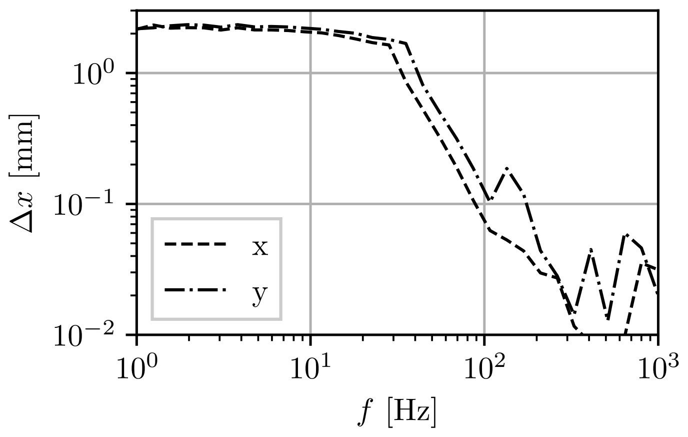


Figure 4.24: Mechanical response of the PP2 sensor head for both axes. Δx is the peak-to-peak distance that the sensor head travels when being excited with a square wave signal for frequency f. Until $f \approx 32\,\text{Hz}$ the sensor head swings over the full range of $\Delta x \approx 20\,\text{mm}$. At $100\,\text{Hz} \lesssim f \lesssim 200\,\text{Hz}$ Δx drops below 1 mm. The frequency at which the movement goes below 100 µm is about 400 Hz.

- The partial HDD-cabinets that are used to hold the magnets and actuators in place limit the access to the actuator arms.
- The long carbon fiber arm extensions induce a lot of angular momentum and therefore reduce the actuator acceleration.
- As the operating range of the optical sensor has been evaluated, a simpler and therefore more compact and light-weight circuit can be implemented on the sensor head.
- The cardboard sheets that hold the `AS5600` evaluation boards in place above the rotating magnets are unstable and spatially and visually limit access to the voice coil mechanics below.
- Since the functionally necessary signal processing components have been identified, the analog signal chains can aggregately be simplified and specialized.
- Relevant functional components are scattered across multiple PCBs, introducing wiring complexity and potentially corrupted signal integrity.

4.6.1 Hardware improvements

In order to overcome and respond to the above listed points, a second revision of the PlayPen has been designed. In this second version of the robot, the *PlayPen-2*, or *PP2* for short, a few major design modifications and amendments were introduced. The most important changes are again listed pointwisely below.

- The signal pre-amplification remains analog but digitally configurable. This allows experiments to be quickly initialized and reliably reproduced.
- Advanced signal processing and spike generation is implemented digitally to further improve reproducability and flexibility.
- To have a larger margin for the code execution, an `STM32F103C8` 32-bit microcontroller (STM, 2015) running at 72 MHz replaces the `Arduino Nano`. This

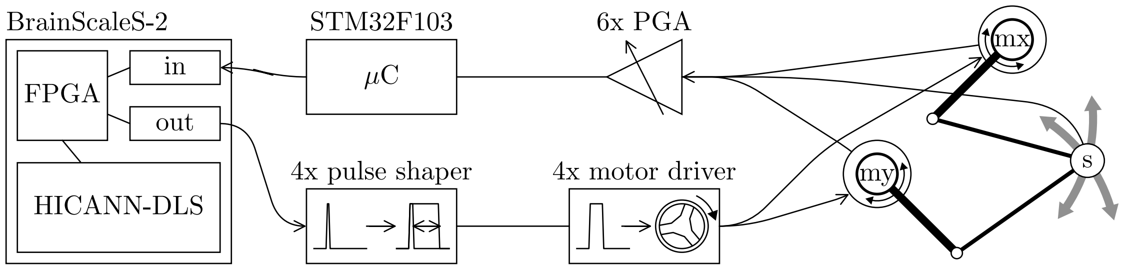


Figure 4.25: Block schematic of the PlayPen-2 signal path. Spikes from BrainScaleS-2 reach the pulse shapers which are set to a fixed pulse width that results in an optimal actuator response. Four motor drivers enable bidirectional movement of each actuator. Signals are obtained from the rotary sensors at the actuator (mx and my) and the optical sensor head (s) and reach the programmable gain amplifiers (PGAs) the return path. Thereafter, the analog signals are digitized and translated into spike trains by the `STM32F103C8` microcontroller and sent into the spike input of BrainScaleS-2.

microcontroller is also carried by a compact development board with its own power supply and USB functionality. Therefore, it can easily be replaced or upgraded for future PlayPen experiments.
- Both actuator coils are driven by one `DRV8833` dual H-bridge motor driver (Tex, 2015a) with current limitation and thermal shutdown.
- The `TXB0104/8` self-configuring level shifters were replaced by externally configured `SN74LVC2T45DCT` transceivers (Tex, 2017).
- The `AS5600` rotation sensors are mounted on small custom PCBs above the actuator sensor magnets.
- The voice coil actuators and the corresponding magnets are mounted directly on the PlayPen PCB.
- The sensor head and its guidance rods are lighter and more stable than before.
- The supply voltage changed from 5.0 V to 3.3 V.

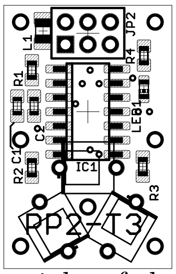
The mechanical actuator responses of the new sensor head to various driving frequencies is shown in figure 4.24. The maximum usable frequency is approximately 300 Hz, above which the peak-to-peak range goes down to below 0.3 mm. According to equation (4.8), the sensor head acceleration is therefore

$$a_{\max} = 4\pi^2 \cdot \Delta x \cdot f^2 \approx 1000\,\mathrm{m\,s^{-2}}. \tag{4.15}$$

Figure 4.25 shows the block schematic of the second revision. The image at the beginning of this section shows the PCB layout. Part of the output path are only four pulse shapers that are set to a fixed pulse width and four motor drivers. Instead of the analog signal processing components in the return path, PP2 only contains six `PGA112` programmable gain amplifiers (PGAs) (Tex, 2015c) and the microcontroller that digitizes the signals with its integrated 12-bit ADCs.

The microcontroller is also responsible for communication with the peripheral units, i.e., the PGAs, a `MCP4922` DAC (Mic, 2010) for offset adjustments in the signal path, and various custom circuits for spike triggering and gating. The complete schematic can be found in appendix C.7.

4.6.2 Sensor head



With the transition from the prototype to PP2, also the sensor head has been updated. From previously only one, the number of photo diodes increased to three. Figure 4.26 shows the sensor head cabinet without the PCB. The cabinet connects the sensor PCB to the guiding rods and shields the photo diodes from disturbing ambient light. Moreover, it guides the relevant light through the small quadratic openings in the bottom and the pyramid-shaped apertures. The total weight of the sensor head including the push rods is 4.6 g.

The `MAX4206` logarithmic transimpedance amplifier has been replaced by three linear transimpedance amplifiers. To save parts and complexity, the circuits are kept very simple and centered around an `AD8648` four-channel rail-to-rail operational amplifier. The `AD8648` has low noise characteristics and an input bias current of only 1 pA to not interfere with the small photo diode currents. The complete schematic can be found in appendix C.9. With the photo current of equation (4.11), the output voltage of one transimpedance circuit is

$$V_{\text{sensor}} = R \cdot I_{\text{diode}} = R \cdot 11 \cdot P^{1.08} \approx 3.6 \times 10^6 \cdot P^{1.08}\,[\text{V}]. \tag{4.16}$$

Here, $R = 330\,\text{k}\Omega$ is the feedback resistor, I_{diode} is the diode current, and P the integrated light intensity over the photo-active area in watts. As currents of up to $I_{diode} = 9.4\,\mu\text{A}$ are expected (see section 4.3.2), the output voltage should be in the range of V_{sensor} =0 V to 3.1 V. Since the supply voltages of all components including ths PGAs on the PP2 setup are 3.3 V, this optimally covers the maximally allowed input range.

The get an estimate for the smallest measurable currents, the smallest reliably measurable signal is assumed to be on the order of a few LSBs that the `STM32F103C8` ADCs can resolve. At at 3.3 V input range and a resolution of 12 bit, the LSB voltage equivalent is $V_{\text{LSB}} = 806\,\mu\text{V}$. If three to four LSBs are assumed to be what can reliably still be resolved in a single measurement, the lowest detectable voltage is on the order of 3 mV, which translates to a smallest measurable current of 9 nA.

Thus, the range is already sufficient to cover all applications that are expected from the considerations in section 4.3.2 but it can further be extended by employing the PGAs. For example, if the light conditions are lower than expected, the PGAs can amplify the signal by a factor of up to 128.

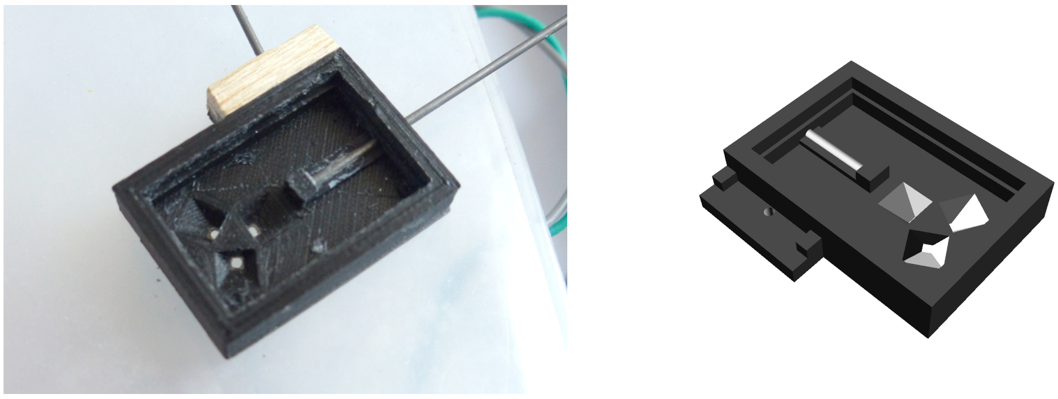


Figure 4.26: Photo and rendering of the cabinet of the PlayPen-2 sensor head without PCB. Visible are the channel for the solidly mounted push rod, the hole for the flexibly mounted push rod, and the three pyramid-shaped aperture channels. The cabinet is 3D-printed

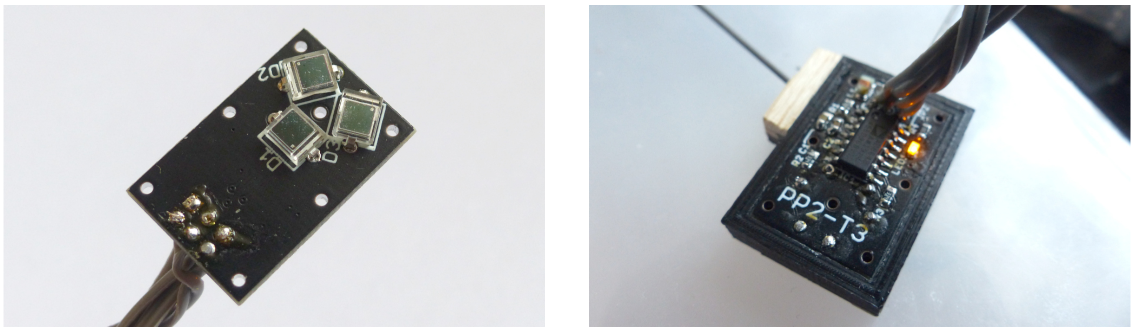
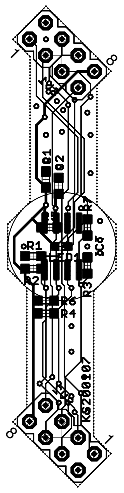


Figure 4.27: PlayPen-2 sensor head PCB. On the bottom side are the three `BPW34` photo diodes and the top side hosts the preamplifiers.

4.6.3 Rotary sensors



The design of the `AS5600` rotary sensor PCBs has been mostly adapted from the evaluation board (ams, 2014) and adjusted to fit the special requirements of PP2. Each `AS5600` is mounted on the bottom side of a bridge-like board that has two eight-pin header connections on each side, which also serve as the mechanical mounting points. The corresponding connections on the PP2 main PCB are to the left and right of the center of rotation of each voice coil actuator. The pinout of the both connections is point symmetric, such that the PCB can be rotated by 180° without being reversed in polarity. The height of the sensors can be adjusted by trimming the length of the pins strips. Figure 4.28 shows this arrangement.

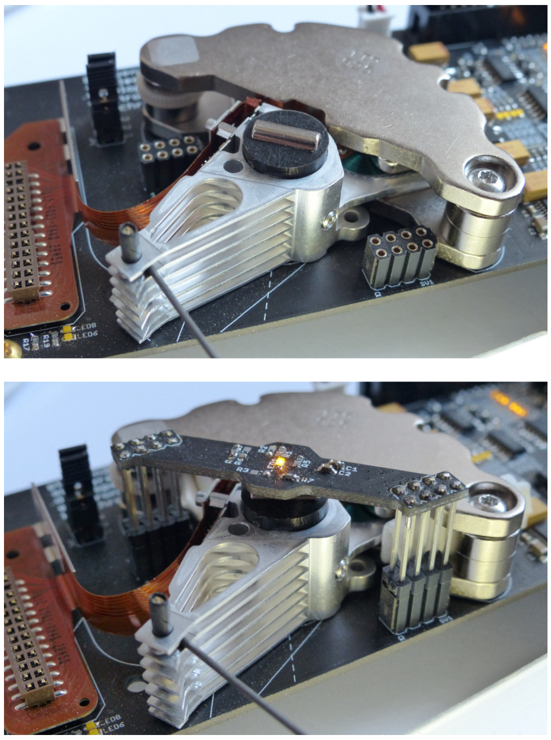


Figure 4.28: PlayPen-2 rotary sensor. Top: The neodymium magnet is mounted in a 3D-printed plastic disk that is, in turn, solidly connected to the aluminum actuator. Bottom: The rotary sensor PCB is centered above the magnet and held in place by the two 2×4 pin strips.

4.6.4 Microcontroller firmware

The complexity reduction in terms of removing most components of the analog signal processing chain is compensated by digital signal processing which is implemented in `C` code (Kernighan et al., 1988) and executed on the microcontroller. For the sake of brevity, only the exceptional and relevant aspects of the software implementation are outlined.

The ADC results are read out and transferred to static memory locations at a rate of 100 kHz. As this transfer is handled by one of the `STM32F103C8`'s direct memory access (DMA) engines in the background, the user routines can seamlessly access an ADC result with a single-cycle instruction from the memory locations and therefore at high performance.

Since the microcontroller has to be able to send spikes at a rate of up to 100 kHz, the software spike generators have to be implemented efficiently. To achieve this, the spike sending functions are called from a timer-triggered interrupt routine that is invoked at a rate of $f_{\text{interrupt}} = 100\,\text{kHz}$. Every call iterates the state of all spike generators. Below is a slightly simplified version of the interrupt routine.

```
__attribute__((optimize("unroll-loops")))
void timerInterrupt(void) {
    // Update spike generator states
    for(uint8_t i = 0; i < NR_SPIKE_GENS; i++) {
        // Takes ~26 clock cycles for each (~360 ns)
        spikeGenHandlePhaseIncrement(&spike_generators[i]);
    }
    // Reset spike pins
    for(uint8_t i = 0; i < NR_SPIKE_GENS; i++) {
        spikeGenResetSpikeOutput(&spike_generators[i]);
    }
}
```

The total number of spike generators, `NR_SPIKE_GENS`, is in the current implementation set to 7. However, the number of active spike generators can by configured dynamically during runtime. The function attribute `(optimize("unroll-loops"))` tells the compiler to not actually iterate over the spike generators but to instead explicitly execute the loop content consecutively. I.e.,

```
for(uint8_t i = 0; i < NR_SPIKE_GENS; i++) {
    spikeGenHandlePhaseIncrement(&spike_generators[i]);
}
```

will be compiled as

```
spikeGenHandlePhaseIncrement(&spike_generators[0]);
spikeGenHandlePhaseIncrement(&spike_generators[1]);
spikeGenHandlePhaseIncrement(&spike_generators[2]);
spikeGenHandlePhaseIncrement(&spike_generators[3]);
spikeGenHandlePhaseIncrement(&spike_generators[4]);
spikeGenHandlePhaseIncrement(&spike_generators[5]);
spikeGenHandlePhaseIncrement(&spike_generators[6]);
```

This increases the code size but safes execution time, which is the far more critical parameter to optimize in this case.

Every spike generator holds a few state and configuration variables. Most importantly, its phase `phi` and its phase increment `delta_phi`. `phi` is increased by `delta_phi` in every call of the interrupt routine until it crosses 1024. On that event, a spike is triggered and `phi` is wrapped back:

$$\texttt{phi} \leftarrow \texttt{phi} - 1024 \tag{4.17}$$

Therefore, `phi` behaves similar to a watch hand that rotates with a frequency defined by the phase increment and the interrupt rate.

$$f_{\text{spike}} = \frac{\texttt{delta\_phi}}{1024} \cdot f_{\text{interrupt}} \tag{4.18}$$

After every full cycle it triggers a spike. Below are the relevant functions.

```
inline void spikeGenHandlePhaseIncrement(spikeGenerator *sg) {
    // Increment phase
    sg->phi += sg->delta_phi;
    // Handle a spike event when we cross phi_max and wrap phi
    if (sg->phi >= 1024) {
        // Send spike if the generator is enabled
        if(sg->enable) {
            // Set the corresponding hardware pin
            GPIO_SetPin(sg->spike_port, sg->spike_pin);
        }
        // Wrap phi
        sg->phi -= 1024;
    }
}
```

```
inline void spikeGenResetSpikeOutput(SpikeGenerator *sg) {
    // Reset the corresponding hardware pin
    GPIO_ResetPin(sg->spike_port, sg->spike_pin);
}
```

The keyword `inline` tells the compiler to not actually call the function by executing a jump to the corresponding program location, but to implement the content of the function at the location where it appears. Again, this has a negative impact on the code size, as the function's content is redundantly implemented at every call, but it safes execution time because the function calls are avoided.

The total execution time of the interrupt routine fluctuates with the state of the spike generators but takes maximally around 3.5 µs, causing a maximal load of about 35 %. The rest of the code can therefore be executed with a lower priority during the remaining 65 %.

After initialization, the system enters an infinite loop from which it monitors and orchestrates all tasks that are less time critical. One loop cycle usually takes 500 µs but can take longer if either the system load is too high or if the system is actively communicating over USB. During each cycle, the buffered ADC results are converted into spike rates. Hence, the sensor unit update rate is 2 kHz, or twice a second in biological time.

Moreover, all ADC traces can be recorded on the microcontroller at the same rate. The trace length is limited by the remaining random access memory (RAM) resources that are not already in use by the program execution. It is statically set to 12 kB. The number of recorded samples is automatically adjusted according to the number of channels that are selected for read-out.

Apart from the sensor-to-spike translation and the ADC recorder, the `STM32F103C8` is also responsible for managing the entire state of the PP2 system and for communication with the host PC. The latter takes place over an USB interface that appears as a serial port on the computer. Due to the powerful periphery of the `STM32F103C8`, the interface can be configured to a data rate of $\sim 2\,\mathrm{MB\,s^{-1}}$, providing generously dimensioned access to the robotic system. Information gets exchanged via a simple text-based communication protocol. However, in order to have a more convenient way to interact with the system, a custom Python library that contains an abstracted representation of the robotic system has also been developed.

The library provides a class called `PP2` with abstract representations of the various robot subcomponents, each of which can further be configured by properties that are hierarchically nested. The main components are

- `PP2.motor_drivers`
- `PP2.dac`
- `PP2.pga`
- `PP2.hdds`
- `PP2.spike_generators`
- `PP2.recorder`

`PP2.pga`, `PP2.hdds`, and `PP2.spike_generators` inherit from lists and can hence be iterated. Every parameter can be directly assigned a value. For example, to enable the motor drivers, the corresponding parameter is set to `True`:

```
pp2.motor_drivers.enable = True
```

After configuring the components like this, the state of the virtual Python object is transferred to the hardware, by calling the `.set()`-method of this component:

```
pp2.motor_drivers.set()
```

Calling the `.set()`-method of a `PP2` instance, calls the `.set()`-methods of all the subcomponents and therefore transfers the state of the entire `PP2` object to the hardware. Parallel to the `.set()`-methods, also `.get()`-methods exist to synchronize the software objects with the robot. The following example illustrates how PlayPen-2 can be configured and used from within Python.

```
# Get the serial interface over which PP2 is connected
port = get_pp2_port()

with serial.Serial(port, 2073600, timeout=1) as s:
    # Create a PP2 instance
    pp2 = PP2(s)

    # Calibrate the HDD sensors
    pp2.calibrate_hdd_sensors()

    # Enable spike generators and assign a channel to each
    for i in range(PP2_NR_SPIKE_GENS):
        pp2.spike_generators[i].enable = True
        pp2.spike_generators[i].adc_channel = i
    pp2.spike_generators.set()

    # Configure the ADC recorder
    pp2.recorder.enable = True
    pp2.recorder.nr_channels = 5
    pp2.recorder.adc_channels = [0, 1, 2, 4, 5]
    pp2.recorder.set_config()

    # Start the recording, wait for 2 seconds, then get the data
    pp2.recorder.start()
    time.sleep(2)
    pp2.recorder.get_data()
    data = pp2.recorder.data
```

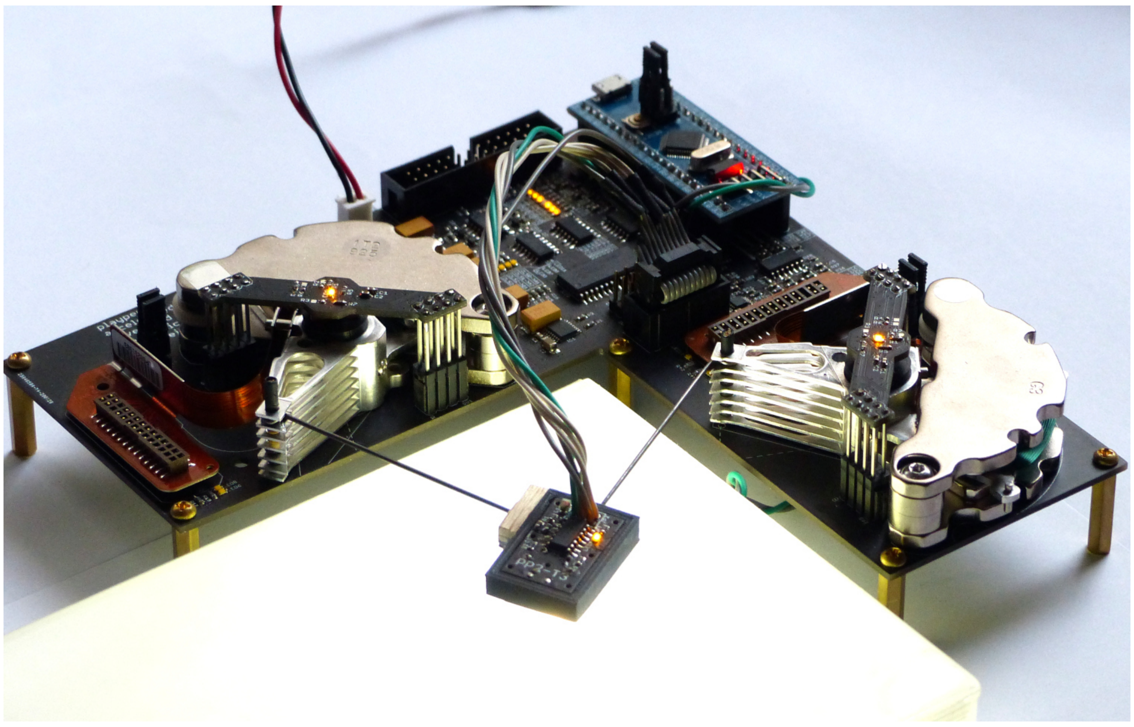


Figure 4.29: The complete PlayPen-2 setup.

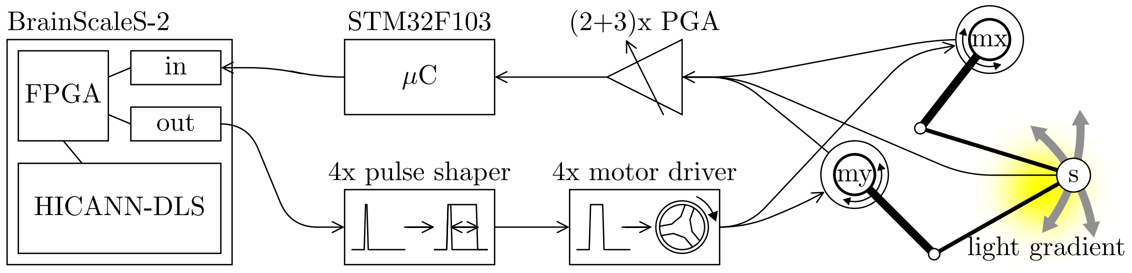


Figure 4.30: Experiment schematic and data flow for maximum brightness tracing with a three-dimensional sensor on PP2.

4.7 Maximum finder on second revision

The goal of this experiment is, again, to find and trace the brightness maximum of an illuminated surface. However, since the upgraded sensor head is able recognize two-dimensional gradients, the network can be structured in a much simpler way.

Figure 4.30 shows the signal path for this experiment in terms of included hardware resources. BrainScaleS-2 sends the output spikes of four motor neurons to four pulse shapers which are directly connected to four motor drivers. The motor neurons can therefore control the motion of both actuators, each in both directions. The angular actuator positions leave the two `AS5600`s as analog signals and are buffered by two PGAs before they reach the microcontroller's ADC inputs. All three preamplified photo signals are also distributed to three PGA channels and then digitized by the microcontroller. The rotary information, in this case, has no influence on the neural network but is only recorded to reconstruct the sensor head positions in the ensuing experiment analysis.

The photo signals are each converted linearly into an output rate

$$r_{\text{out}} = r_{\text{max}} \cdot \frac{V_{\text{signal}}}{V_{\text{max}}} \tag{4.19}$$

with $r_{\text{max}} = 100\,\text{kHz}$. V_{max} is calibrated such that at the brightest point $V_{\text{signal}} = V_{\text{max}}$. Thus, each photo sensor is assigned a separate spike generator and a separate spike source channel. After reaching BrainScaleS-2, the spikes are distributed over three synaptic rows, in particular, one row for each sensor.

Some synapses within these rows are assigned a discrete weight such that they cause a non-zero postsynaptic potential on the synaptic input of the respective motor neurons after receiving a spike. In turn, this might cause the receiving neurons to fire. Each motor neuron is statically assigned one actuator direction and therefore exerts a torque pulse every time it spikes.

Figure 4.31 illustrates this chain of effects in a more graphical way.

In each individual experiment the neural network assumes control of the movement patterns for about 150 ms, which corresponds to 150 s in biological time. Before an experiment starts, the sensor is externally forced to an arbitrary boundary location by the control software, such that the system has to find the maximum from a different starting position in each run.

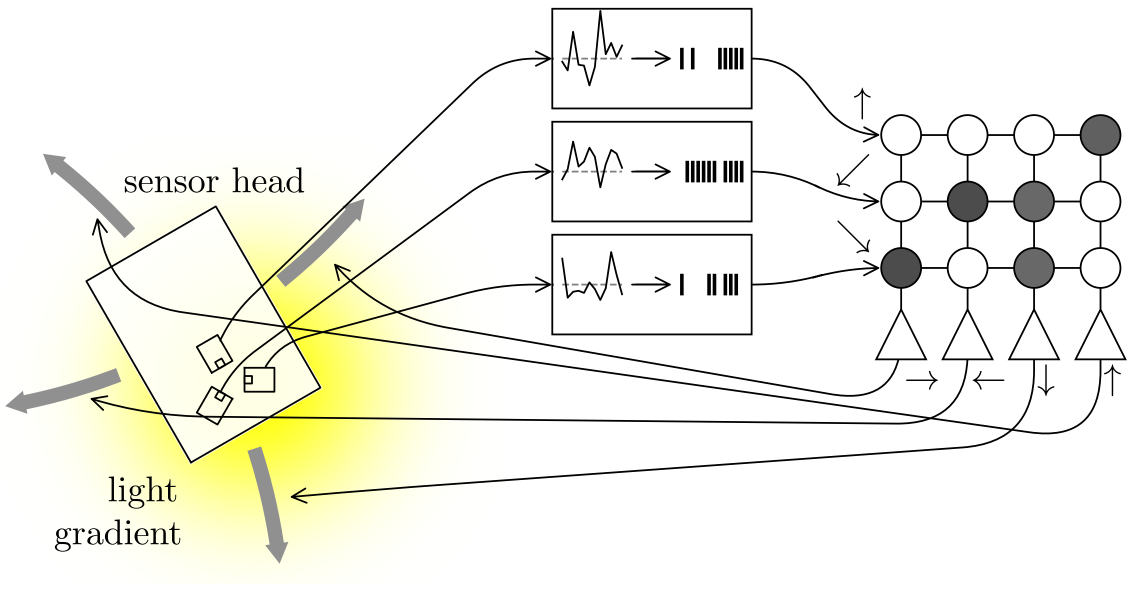


Figure 4.31: Detailed data flow and network architecture for brightness maximization.

In figure 4.32, some return journeys are plotted. The gray shape in the background indicates the area that is reachable by the sensor head. Each of the four trajectories starts from a different position in one of the corners. The reason why some of them do not seem to start exactly in the corner is because there is a latency between the signal that forces them to the start position and the start of neural control. Moreover, the flexible PCBs (the golden-brown bands on the left of the actuators in figure 4.28) exert a force on the actuators that is insignificant when the actuators are actively driven but causes them to drift away from the edge position when no coil currents are applied. Therefore, three of the four edge locations are unstable when the actuators move passively. However, the procedure is sufficient to assert a different starting position for each run.

After the active control starts, the sensor head is driven to a point around $x = 11.5\,\mathrm{mm}$ and $y = 12.7\,\mathrm{mm}$. The actuators forcefully move the sensor into the approximate region of this point, also including some overshooting. After that the target location is approached more slowly and more precisely.

Figure 4.33 shows the location and sensor signals of the blue trajectory from figure 4.32 over time. At $t = 100\,\mathrm{ms}$ the neural network assumes control over the actuators and it stops at $t = 250\,\mathrm{ms}$. The gray curves in the sensor plots show the average over the three sensor signals, which is what the network is intended to maximize. While it increases by tendency, the value at $t = 180\,\mathrm{ms}$ is higher then the final value. Hence, the system does not converge the absolute maximum. This is due to the fact that the weights have not been optimized but manually tuned until the network achieved its goal qualitatively.

In the next experiment, the parameter tuning is automized by an evolutionary strategy algorithm.

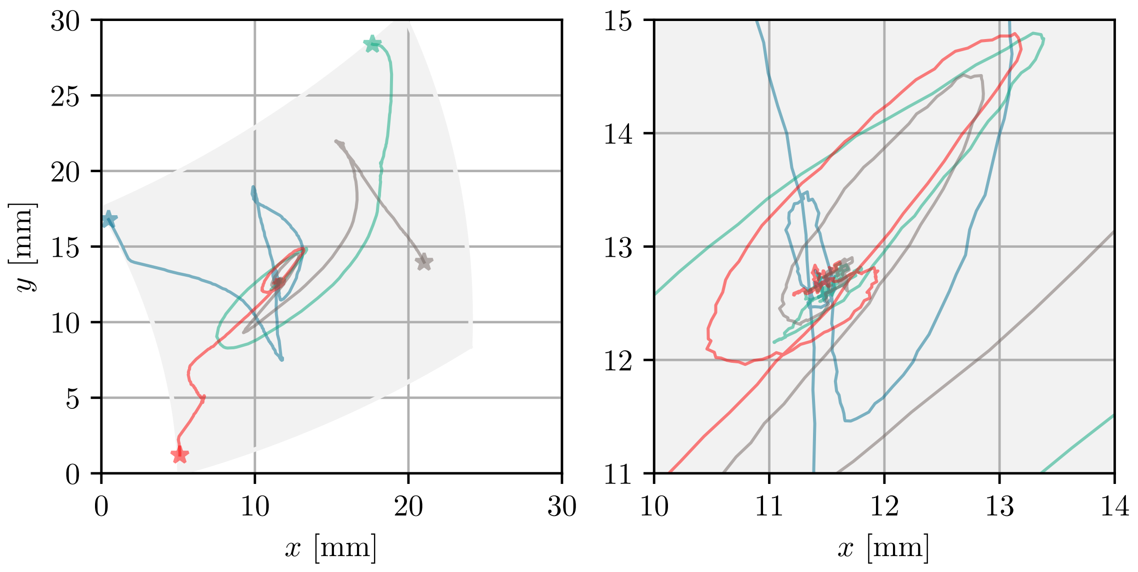


Figure 4.32: Trajectories of the sensor head starting from four different positions. Left: The gray shape in the background marks the area that is accessible to the sensor. The stars indicate the respective starting positions. Right: Zoom into the region of the brightness maximum.

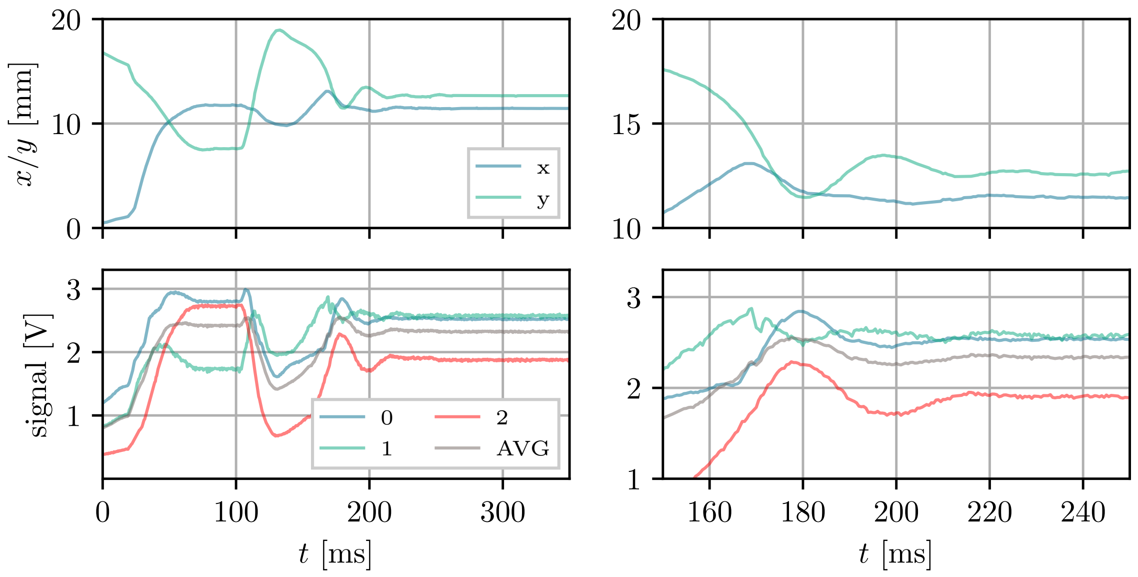


Figure 4.33: Position (top) and optical sensor (bottom) signals over time. Left: All signals are shown from 0 ms to 350 ms. The neural network takes control at $t = 100$ ms and stops at $t = 250$ ms. Right: A zoom into 100 ms to 200 ms. The gray curve in the lower plots is the average over the three photo diode signals.

4.8 Evolutionary optimization

In this experiment, the task is again to find the brightness maximum but the parameters were further optimized by an evolutionary algorithm, in particular an evolution strategy with covariance matrix adaptation (Hansen and Ostermeier, 1996).

The algorithm has control over the same twelve weights as depicted in figure 4.31, and optimizes a fitness function that it derives from the three sensor signal $S_s(t)$. In order to prevent artifacts that might arise from unfair starting conditions, the fitness of one set of weight parameters is always derived from averaging over four runs from four different starting positions as in figure 4.32. Thus, the three sensor signals get averaged over the sensor index s, the time t, and the four starting positions r:

$$f = \frac{1}{4 \cdot 3 \cdot t_{\text{stop}}} \sum_{r \in \{0,1,2,3\}} \sum_{s \in \{0,1,2\}} \sum_{0 \le t < t_{\text{stop}}} S_s(t)|_{\text{run}=r}. \tag{4.20}$$

f is therefore proportional to the average brightness received by the three diodes. Moreover, since the runtime is finite and since the averaging starts at $t = 0$, f also reflects how fast the sensor approaches maximum brightness.

Initially, 50 weight vectors $\boldsymbol{w}_i$ are each drawn from a Gaussian distribution whose mean vector corresponds to the weight vector from the previous example from section 4.7 and whose standard deviation is set to 3.0 for all weights. A weight vector $\boldsymbol{w}_i$ is also referred to as an *individual* in reference to evolutionary algorithms. After all individuals with index i of this first generation have been evaluated and assigned a fitness f_i, the best five are selected as the parents of the next generation.

In the following, the next 50 weight vectors are then drawn from a multivariate Gaussian distribution whose mean is obtained from the fittest individuals and whose covariance matrix adapts to the apparent success of the individual genes:

$$\boldsymbol{w}_i \sim \mathcal{N}(\boldsymbol{\mu}, \sigma \cdot \boldsymbol{\Sigma}). \tag{4.21}$$

Here, $\sigma = 3.5$ is a heuristically chosen step size and

$$\boldsymbol{\Sigma} = \sum_i p_i \boldsymbol{d}_i \otimes \boldsymbol{d}_i \tag{4.22}$$

is the covariance matrix with $\boldsymbol{d}_i = \boldsymbol{w}_i - \boldsymbol{\mu}$ and $\boldsymbol{\mu} = \sum_i p_i \cdot \boldsymbol{w}_i$. Furthermore, $p_i = \frac{1}{5}$ if the individual i is among the five fittest and $p_i = 0$ otherwise. Scaling the covariance matrix in this way tends to warp the variance into directions that promise successful mutations. The weights are allowed to be in the interval $w_{i,j} \in [0; 50]$. If equation (4.21) produces a weight that is not within this range, it is clipped to the respective range boundary. The maximum weight is set to 50 instead of technical maximum 63 in order to limit the impact one presynaptic partner can have on a single neuron and to have a certain tuning margin in case the stimuli are generally found to be too weak to trigger sufficient motor neuron responses. Typically, the optimization converges already after approximately 20 generations.

Figure 4.34 shows the course of this evolution for a few selected generations starting from the top left and going to the bottom right, rowwise. The background

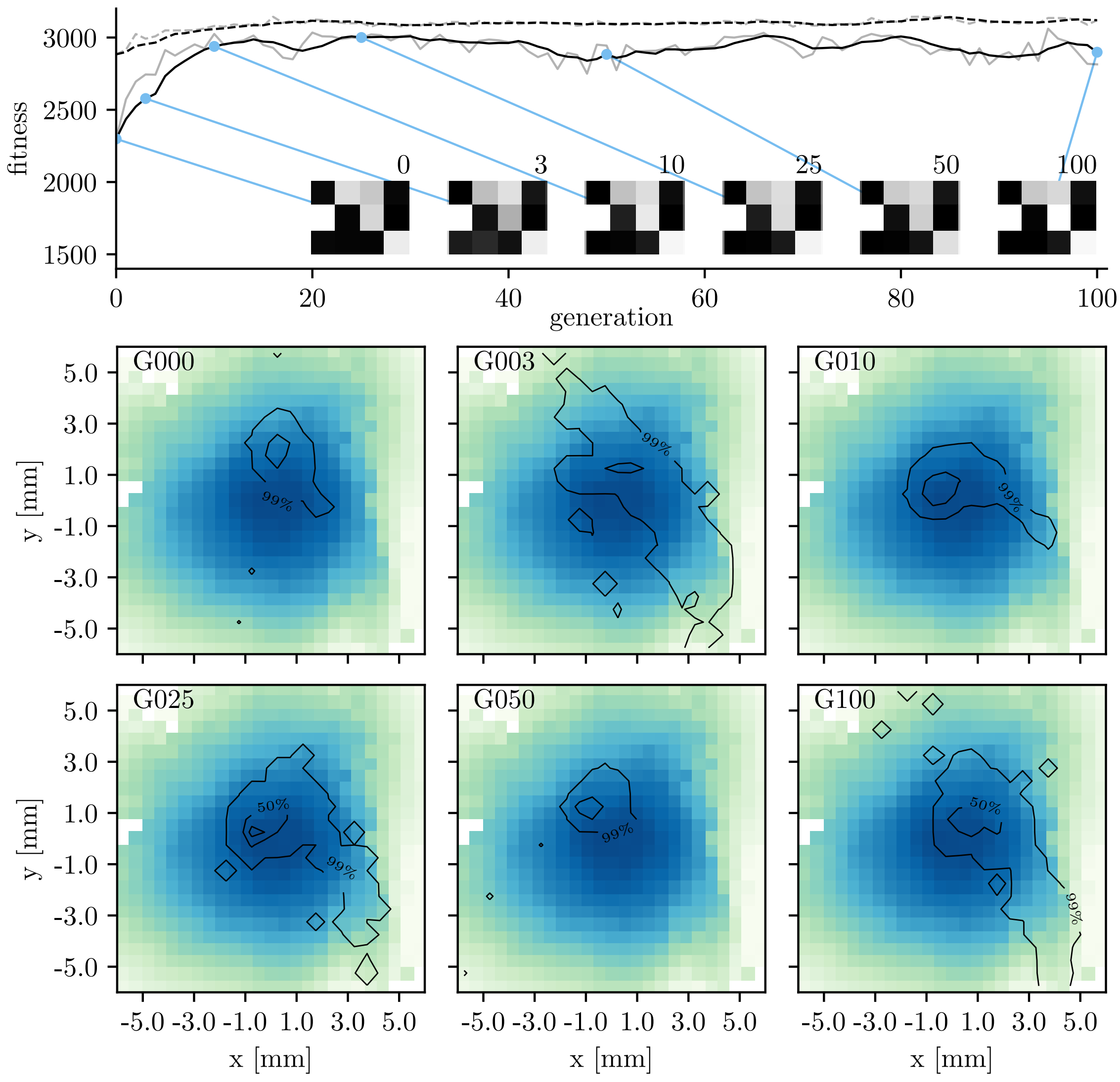


Figure 4.34: Evolution of the maximum finder starting from a heuristically chosen initial weight configuration. The dark solid curve in the top plot is the average fitness of the evaluated population. The light solid curve is the corresponding moving average over five samples. The dashed curves are the fitnesses of top three individuals, again the value per generation and the moving average. Additionally, the weight vector $\boldsymbol{\mu}$ is plotted qualitatively at six selected generations (0, 3, 10, 25, 50, and 100). The mean weight vectors of the same six generations were evaluated for their performance by conducting 100 experimental runs, each subdivided into four runs from four starting positions (see figure 4.32). The contour lines in the lower six tiles indicate the distributions of the sensor head positions after 225 ms, when the target positions should be reached. The outer contours contain 99 % of the samples, the next inner contours, 50 %, and the innermost contours 10 %. The background color represents the sensor brightness obtained from sampling the photo diode signals over all runs and all generations. Some pixels are missing because the sensor head did not cross the respective positions.

of each tile represents the brightness obtained from sampling the sensor information from all conducted runs. The black contours reflect the probability of presence of the sensor head after 225 ms when the position is usually settled.

In generation 0, the trajectories are localized at $x/y \approx 0.0\,\mathrm{mm}/2.0\,\mathrm{mm}$, a position that is not in the very center of brightness, but at a gradient slightly upwards of it. As the optimization proceeds, the distribution of target locations continues to diffuse and narrow again but its center moves toward brighter areas, thus, achieving its objective. However, overall not much changes, indicating that the heuristically chosen parameter set from section 4.7 is either already close to optimal or leads into a local minimum. Moreover, the distribution seems to avoid the peak brightness while elongately lingering around areas up and right of the brightness maximum.

In order to test, whether different starting conditions produce different parameter sets, the evolution is conducted a second time starting from a totally random weight configuration. The first generation is made up of individuals whose weights are each drawn from a uniform distribution within the allowed weight ranges, i.e. $w_{i,j} \in [0; 50]$. Figure 4.35 shows again the target location probabilities for a few selected generations. The network successfully reaches a point of maximum brightness and it converges into a functional state after approximately 20 generations. This time however, the absolute fitness value is higher and the distribution remains narrow and circular, covering the very center of brightness. A look at the weight matrices reveals that the final weight configuration is different from the one obtained with heuristically chosen starting conditions. Hence, the evolution with artificially set initial conditions indeed ended up in a local minumum.

One evolution, as presented above, took approximately 2.5 hours. During this time, 100 generations of 50 individuals each performed four test runs. Thus, the robot carried out 20000 evaluations in total for one evolution. Translated into biological real-time considering the acceleration factor, the system would have had to run for three months instead.

4.9 Outlook

The PlayPen setup allows accelerated neuromorphic hardware to be interfaced with physical robotic actuators and sensors in a low latency, closed-loop fashion. Experiments with the prototype setup (sections 4.3 to 4.5) revealed a few weaknesses, which for the most part were eliminated in the second revision. Furthermore, the software representation of PP2 facilitates easy, stable, and reproducible configuration of and interaction with the robot. In this way, it becomes possible to automate complex experimental sequences such as those required for training with evolutionary algorithms.

One remaining aspect, which could not be satisfied according to the original objectives, is the mechanical acceleration capacity of the actuators. Regulation oscillations are still on the order of 10 ms, which is one to two orders of magnitude below what would correspond to a typical biological movement translated to the

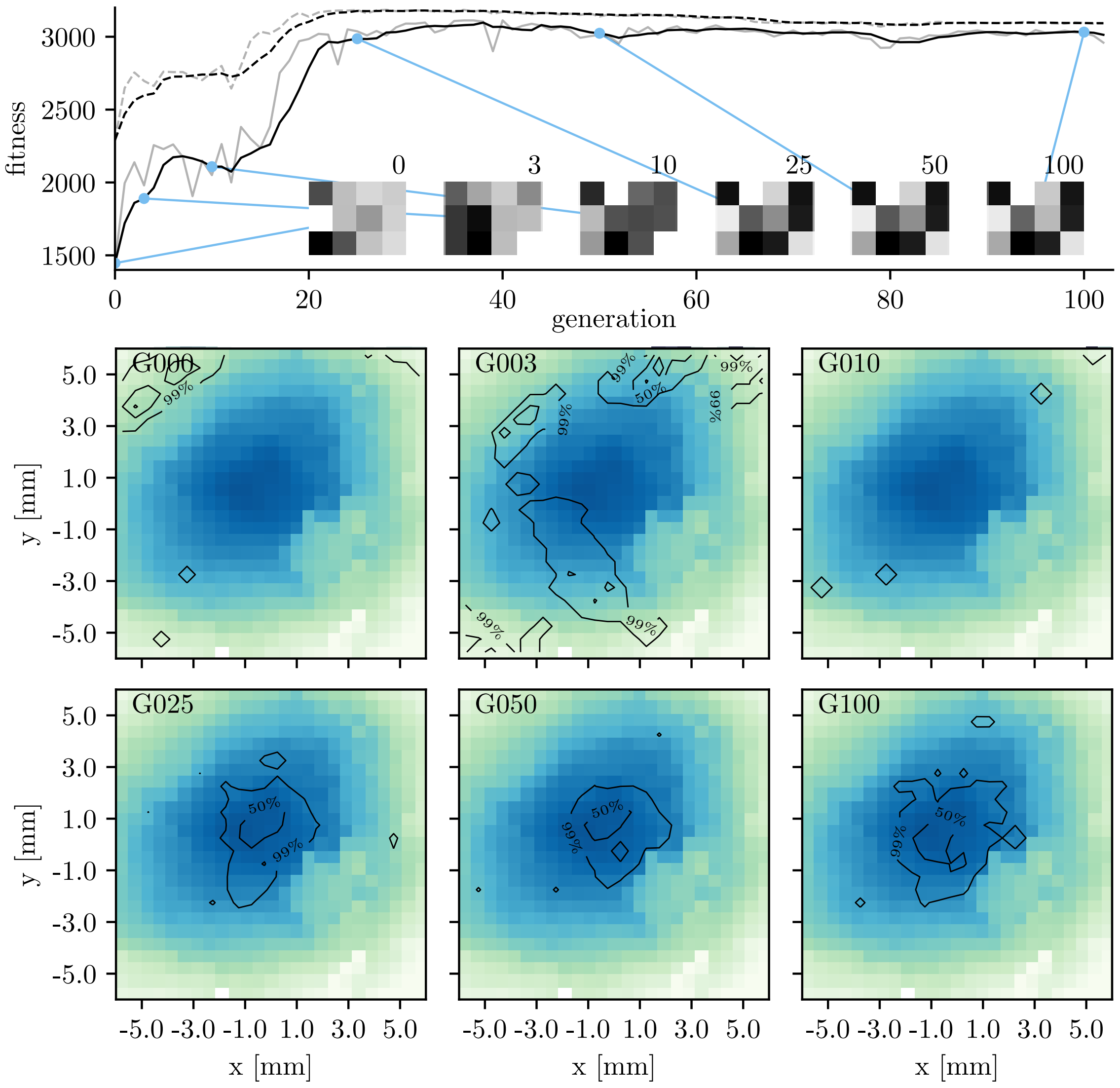


Figure 4.35: Evolution of the maximum finder starting from a random initial weight configuration. The reader is kindly referred to the caption of figure 4.34 for an elaborate detailed description of the figures at hand.

thousandfold BrainScaleS-2 speedup. More electrical power and a further reduction of mass in the actuation complex could help meeting the demands but the scaling relation $P \propto \frac{m}{\tau^3}$ (equation (4.2)) suggests that both, mass and power, have only linear impact against the third power of the typical reaction time. Thus, achieving an improvement of one to two orders of magnitude in the dynamics requires significant changes in the system design, including replacing the HDD-based voice coil actuators and preferably dumping the sensor head. For example, robotically controlled sensory access to a two-dimensional surface could also be achieved by mirror-based optical guidance. In this way, the sensors could be statically mounted, reducing the number of moving parts to two light-weight mirrors.

However, since the emulated LIF neurons on BrainScaleS-2 are naturally also capable of operating in the rate limit, the robotic movements are of course accessible to neural control. Moreover, PP2 accelerates typical biological scales by one or two orders of magnitude. Therefore, an actuator action that lasts for 10 ms, can be understood as an action that takes 10 s in the neuromorphic time frame. This duration is not so far from many biological courses of action, like for example, moving through space, or observing a situation. Hence, the PP2 sensor head speed is in an acceptable regime, when interpreted as a moving agent rather than a moving limb.

Thus the range of possible experiments is far from being exhausted. Together with the extended possibilities of the full-scale BrainScaleS-2 version, many more complex application areas emerge. In addition, the neurons on BrainScaleS-2 offer more generous parameter ranges, so that the lack of speed of the system can be compensated for by slower neurons.

In addition to this physical robotic setup, the next chapter presents and experiment with a virtual cybernetic insect.

5 Accelerated insects

"Of course, the point was not to duplicate wasps, flies, spiders, or bees in computer chips or the like; the important thing was their neural anatomy with its built-in sequences of directed behavior and programmed goals."
– Stanisław Lem, Weapon systems of the 21st century or The upside-down evolution, 1983

As outlined in section 2.2, one major motivation for neuromorphic hardware lies in simulations of large-scale brain structures like the neocortex. The reason for this is the immense computational complexity of tens of billions of neurons that greatly surpasses the capabilities of contemporary computing devices. However, among the tiny representatives of brains lie equally unexplored structures with equally interesting properties. One example are the neural compounds that are associated with spatial navigation in insects.

While even the largest insect brains measure only a few cubic millimeters in volume and do not contain significantly more than around one million neurons (Burrows, 1996; Menzel and Giurfa, 2001), the cognitive and behavioral capabilities of insects are impressive and manifold.

For example, various moth species perform nocturnal seasonal migration over distances of more than 1000 kilometers, flying 400 km to 500 km per night at altitudes of up to around 1000 m (Chapman et al., 2002; Adden et al., 2020). Bees can remember multiple spatial locations at once and even communicate the relative directions and distances to fellow bees by means of a symbolic language (Frisch, 1949; Von Frisch, 1967; Collett and Collett, 2002). While it has earlier been shown that bumblebees can learn to distinguish visual cues (Nityananda et al., 2014), a recent study revealed that they are even able to acquire complex behavioral patterns from training. In particular, Loukola et al. (2017) rewarded bumblebees for rolling tiny balls into a target area. The bumblebees did not learn by trial and error but mostly by observing other bumblebees who had already adapted the skill. Moreover, they were able to anticipate the best strategy to obtain quick reward when confronted with more than one ball.

As a result of considerable progress in neurophysiological measurement techniques, more and more insights into the neuronal processes from which the above examples ultimately result are now being gathered. For instance, electron microscopy and image reconstruction have recently made it possible to map and reconstruct the entire connectome of *Drosophila melanogaster* (Chiang et al., 2011; Xu et al., 2020). Based on such studies, deeper insights into the mechanistic processes behind insectoid learning (Takemura et al., 2017), or motion detection (Takemura et al., 2013), could be derived. A recent example also includes a neural model, which can reproduce the ability of bees to return to their nests after foraging (Stone et al., 2017).

This chapter outlines how this model is emulated on the analog core of BrainScaleS-2 and coupled to a simulated body in a virtual environment on the digital plasticity processing unit (PPU). The PPU also implements the sensor and motor complexes that connect the virtual agent to the physical neuromorphic network. In effect, the neuromorphic brain is absorbed into an ultra-low-latency cybernetic loop that perfectly meets the speed requirements of its accelerated neurons. In order to fully understand the model function and terminology, it is vital to start with a short phenomenology of insect brains.

Contribution

The author conceptualized, implemented, and evaluated all experimental aspects of this chapter. Philipp Spilger offered significant consulting and support concerning the PPU compiler. Parts of the calibration routines (sections 5.5.1 and 5.5.2) are based on previous work by Sebastian Billaudelle, Benjamin Cramer, and Yannik Stradmann. Timo Wunderlich contributed to software simulations which preceded the hardware implementation. At the time of writing, it is being prepared for publication.

5.1 The central complex

The brain area that is of specific interest for the presented neural model is the *central complex*, a highly convoluted compound that is located in the very center of insect brains. Fossil evidence suggests that this brain region evolved over a period of at least 520 Mio. years since the arthropods emerged in the Cambrian (Ma et al., 2012). Until today, this area is highly conserved among various arthropod species.

The *central complex* is usually subdivided into four substructures, the *protocerebral bridge*, the *ellipsoid body*, the *fan-shaped body*, and the paired *noduli* (Turner-Evans and Jayaraman, 2016). The *ellipsoid body* is also referred to as the *central body lower*, and the *fan-shaped body* as the *central body upper*. Figure 5.1 shows a three-dimensional reconstruction of the entire compound of the female tropical nocturnal bee *Megalopta genalis*.

The *central complex* is connected to many other brain areas, but most importantly for the topic at hand, to the *lateral accessory lobes*. These lobes are considered premotor control regions, relevant for locomotion, but they also receive input from the *anterior optic tubercles* that process information coming from the *optic lobes* (Heinze and Homberg, 2009; Homberg et al., 2011). Figure 5.2 gives an overview of the entire brain of the same bee with highlights on the mentioned brain regions.

While it has long been speculated that the *central complex* is involved in motor control and visually guided behavior (Müller et al., 1997), more recent studies gave firm evidence for its involvement in orientation and navigation (Neuser et al., 2008; Varga and Ritzmann, 2016; Le Moël et al., 2019). Much of the recent research is centered around a specific navigation strategy known as path integration.

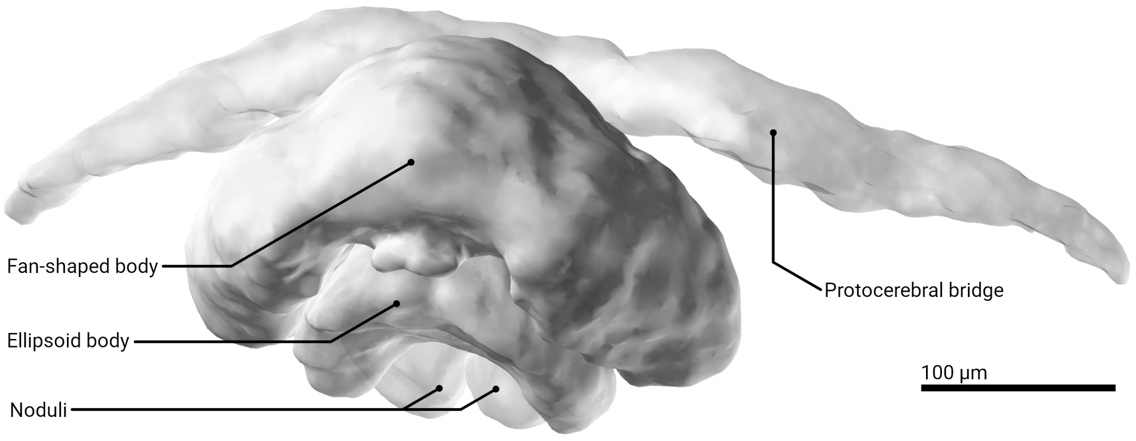


Figure 5.1: 3D reconstruction of the *central complex* of *Megalopta genalis* consisting of the *protocerebral bridge*, the *ellipsoid body*, the *fan-shaped body*, and the two *noduli*. The image was generated on insectbraindb.org based on data from Stone et al. (2017). Labels and the scale bar have been added retrospectively. (IBdb CC-BY)

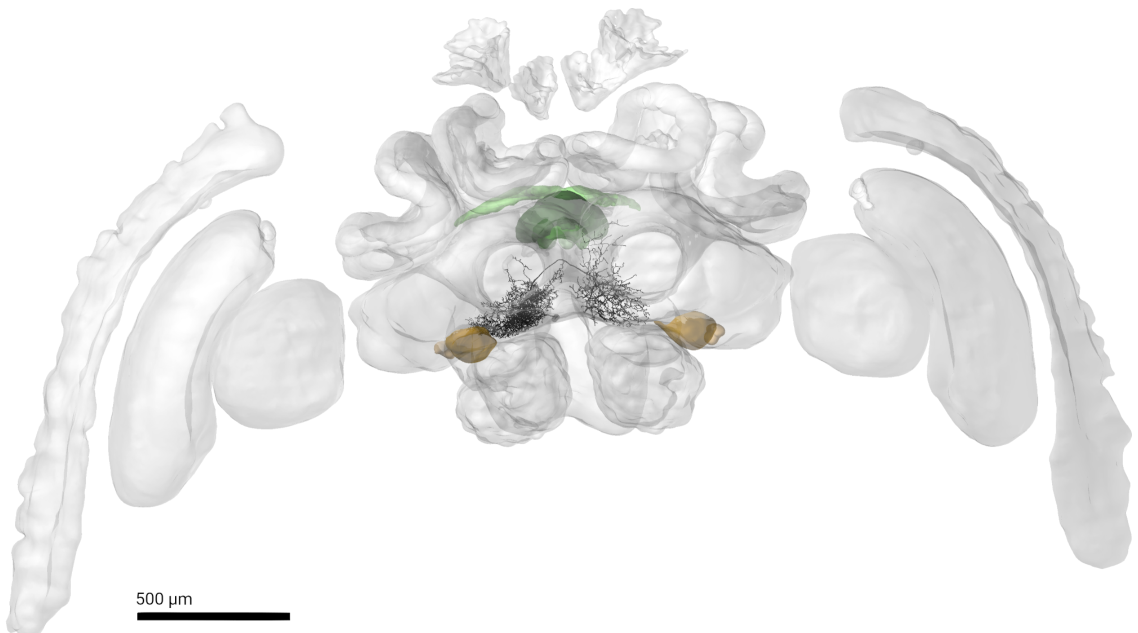


Figure 5.2: 3D reconstruction of the brain of the tropical nocturnal bee *Megalopta genalis* (female). Two brain regions are highlighted in color: the *central complex* in green and the *anterior optic tubercles* in brown. In addition, a neuron that pervades the *lateral accessory lobes* is rendered in black. The large, outer structures (three large ones on each side, the outermost prominently vertically aligned, the innermost round) are the optic lobes that sit behind the eyes and process visual information. The image was generated on insectbraindb.org based on the data set for the neuron mg-Inter-LAL-DN1 (Szakal et al., 2015; Stone et al., 2017). The scale bar was enlarged for better visibility. (IBdb CC-BY)

5.2 Path integration

Path integration, also know as dead reckoning in technical navigation, is a mechanism that infers an object's absolute position by integrating over the history of its locally taken movements. In order to do so, it depends on continuously available direction and speed information during its journey. So called inertial navigation systems in ships or airplanes, therefore rely on compass information, paired with acceleration data or speed measurements. Because deviations from the actual speed or orientation are accumulated and summed up over time, path integration relies on exact measurements and data interpretation.

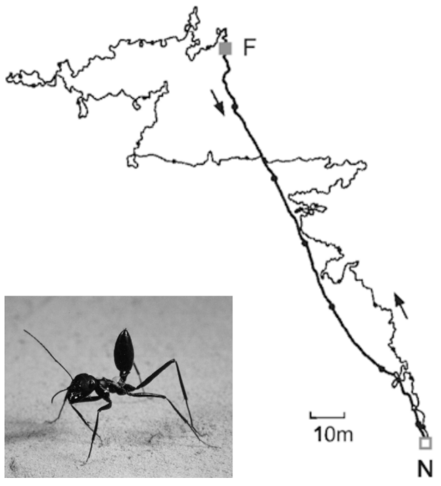


Figure 5.3: Path integration in the Saharan dessert ant *Cataglyphis fortis.* An ant's tortuous outward (foraging) and straight homeward path recorded in a featureless salt pan. *F* denotes the feeding site, *N* the nest. The trajectory data is taken from Wehner (2003) and the image from Möller et al. (2001).

Speculation about similar mechanisms in animals, in particular in insects, goes back at least to the early 20th century (Wehner, 2003). In the late 1940s, experiments gave evidence for the ability of bees to infer compass information from the light polarization in the sky (Frisch, 1949), which they do not only use to navigate but on the basis of which they also communicate food locations to other bees. Later studies have shown that compass information is gathered by various insect species not only via sky light polarization but also by the locations of sun (Heinze and Reppert, 2011) and moon (Warrant and Dacke, 2016), or even the orientation of the milky way in the night sky (Dacke et al., 2013). Experiments with *Drosophila melanogaster* even revealed that the tiny fly is able to keep track of its head orientation for a certain amount of time when optical cues are entirely missing. *In vivo* neural imaging of head-fixed flies shows that they do so by integrating over their own movements (Green et al., 2017, 2018). Interestingly, the obtained results are surprisingly well in line with an early proposed neuronal model for angular path integration (Wittmann and Schwegler, 1995). Like in a compass, the current heading direction is population-encoded in the *protocerebral bridge* and the *ellipsoid body* (Seelig and Jayaraman, 2015). The same encoding is observed in locust (Heinze and Homberg, 2007), butterflies (Heinze and Reppert, 2011), and bees (Stone et al., 2017), as well.

One of the most influential research on insect navigation was centered around *Cataglyphis fortis* (Wehner, 2003; Jeffery, 2003). This ant species is endemic to the Saharan dessert – an almost featureless environment in which the ants navigate without clear-cut visual cues like trees reliably over distances many thousand times their own body length (Huber and Knaden, 2015). Figure 5.3 gives an example of such a recorded ant trajectory over a distance of a few hundred meters. During adolescence when they acquire their extraordinary navigation skills, these ants calibrate their internal compass using the earth's magnetic field as an initial reference and learn to use any available cues (sky light polarization, sun position, memory of movements, the shape of the horizon) to construct more and more reliable compass information (Grob et al., 2019). Besides the directional orientation, path integration requires precise knowledge about the traveled distances. Inferring this distance over time from different kinds of available sensor information is called odometry.

It has been shown that *Cataglyphis fortis* mostly relies on pedometric information, i.e., it counts its steps (Ronacher, 2008). Supporting evidence comes from a prominent experiment in which desert ant received leg modifications before returning to their nest (Wittlinger et al., 2006). Increasing the ant's step size by attaching stilts to their legs, caused the insects to walk in the correct direction to their nest, but past it. When the ant's legs where shortened, they returned into the correct direction too but expected the nest entry too early on their home trajectory. However, the odometric processing that their path integration mechanism relies on, seems to take more information than only the number of steps into account. For example, *Cataglyphis fortis* corrects its number of steps for three-dimensional obstacles that might lie in the return path but not in the foraging path (Wohlgemuth et al., 2001; Grah and Ronacher, 2008; Ronacher, 2008). It can compensate for the additional number steps that are made necessary by slopes. Hence, also for odometry it seems that the insects interweave multiple available sensory cues to increase reliability.

Similar features and behaviors have been found in other insects too. While land bound species tend to rely more on pedometry, flying insects derive information mostly from visual cues and air flow (Wagner, 1986; Srinivasan et al., 1996; Collett and Collett, 2000; Khurana and Sane, 2016). Although the exact mechanisms that lie between the raw optical and sensory information and the odometric representation are yet unknown and subject to ongoing research (Floreano et al., 2009; Yakubowski et al., 2016), neurons can be found in the bee brain which encode different aspects of the optical flow as abstract scalar values. These **t**angential **n**oduli (TN) neurons (Stone et al., 2017) come from the *lateral accessory lobes* and provide input to the *noduli.* Thus, together with the head orientation cells, they potentially deliver all the necessary information for a functional path integrator within the *central complex.*

5.3 A neural model for path integration

The authors of the publication introducing these TN neurons also propose a complete corresponding physiologically derived model that is able to reproduce the observed biological phenomena functionally (Stone et al., 2017). The following presents a neuromorphic implementation that is fundamentally based on it. Following the original publication, the model is discussed below without referencing each point individually. Thus, if not explicitly mentioned otherwise, the reference is Stone et al. (2017).

We start with assigning the individual neural subpopulations to the respective compartments in the *central complex.* For a detailed anatomical reasoning behind the various connections and simplifications, the reader is kindly referred to the original publication. Unlike the compact language that physicists tend to use to

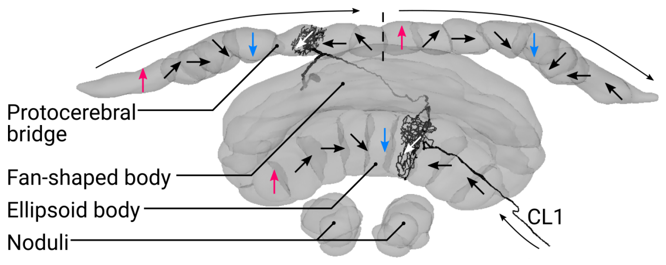


Figure 5.4: Orientation encoding in the central complex of the dung beetle *Scarabaeus satyrus*. The image was adapted from insectbraindb.org, based on data from el Jundi et al. (2018).

describe their aseptic and purified systems, the language used to describe insect brains, despite efforts taken (Ito et al., 2014), is necessarily more complex as it aims at reflecting the given biological complexity in an uncompressed way. The author of this thesis therefore begs the reader, especially the physicists among them, for a certain amount of patience with the terminology. After digging deep into the cell structures of the involved neurons, a simplified model is introduced that tries to put a large complexity back into simple terms.

5.3.1 Subpopulations and network topology

Orientational information is provided by the compass network, which is subdivided into three stages. First, a population of **t**angential **l**ower division of the central body (TL) neurons projects orientational input into the **c**olumnar **l**ower division of the central body, type 1 (CL1) neurons of the *ellipsoid body*. Cell imaging and recording suggests that these TL neurons collect the orientational information from the *lateral accessory lobes* (Homberg et al., 2011). As mentioned earlier, the *ellipsoid body* is known to encode the angular head orientation in a ring-shaped network of neurons. Unlike the *protocerebral bridge* and the *noduli*, the *ellipsoid body* does not have a clear lateral separation into two hemispheres. Instead, like the *fan-shaped body* too, it appears to merge information from the both brain sides (see figures 5.4 and 5.5). The CL1 neurons, in turn, project to inhibitory **t**angential protocerebral **b**ridge, type 1 (TB1) neurons in the *protocerebral bridge*, which has a well recognizable, hemispherically defined activity distribution. It is organized in eight to nine vertically aligned columnar bundles, called *glomeruli*, per hemisphere that each represent a specific heading direction (Turner-Evans and Jayaraman, 2016). *In vivo* recordings show that the two halves of the *protocerebral bridge* each reflect the compass information (Green et al., 2017) from CL1. Schematically, this is illustrated in figure 5.4 for the dung beetle *Scarabaeus satyrus*. The *ellipsoid* and *fan-shaped body* therefore contain only about half as many *glomeruli* as the entire *protocerebral bridge*. In effect, the two TB1 halves also carry the angular information redundantly only with slightly different characteristics.

This information is then coupled into **c**olumnar **p**rotocerebral bridge/**u**pper division of the central body, type 4 (CPU4) cells that stretch through the entire *central complex* and arborize in the *protocerebral bridge*, as well as in the *noduli*, such that they receive excitatory input from both, the TB1 and TN cells. Within the model, the CPU4 population is therefore assumed to implement the actual path integration, since it has both, the speed, and the heading direction available. Thus, CPU4 is assumed to encode the integrated homing vector.

The CPU4 neurons also arborize in the *fan-shaped body*, where they project their activity to **c**olumnar **p**rotocerebral bridge/**u**pper division of the central body, type 1 (CPU1) neurons. These neurons are the largest columnar cells in the *central complex* and are known to converge at the *lateral accessory lobes*, which are, as mentioned earlier, presumably premotoric control regions. They also collect directional input from the *protocerebral bridge* and their structure suggests that each hemispherical population correlates the stored home vector in CPU4 with the heading vector in TB1. However, the way in which the two directions are correlated with each other is literally twisted:

CPU1 is subdivided into two types of neurons, CPU1a and CPU1b. Since the *protocerebral bridge* is laterally elongated and does not appear to be circularly closed like the ellipsoid body, this angular shift is achieved by a structural wrapping from one end to the other. Figure 5.5 illustrates this wrapping with the example of five different CPU1 neurons in the brain of the monarch butterfly. The individual neuronal arborization regions reflect the positions of the columnar *glomeruli* in the *fan-shaped body* as well as in the *protocerebral bridge* and thus, the encoding regions for the different directions. The directions are ordered from the left to the right in both compartments of the bridge such that functionally, the two halves are not mirrored but consecutive copies. I.e., the leftmost *glomeruli* in the left halve of the *protocerebral bridge* represents the same direction as the leftmost *glomeruli* in the right halve (see also figure 5.4). The CPU1a neurons connect the left *protocerebral bridge* compartment with the right *lateral accessory lobe* crossing though the *fan-shaped body.* The very left of the latter is connected to CPU1a R7 which is not at the very left of the *protocerebral bridge* compartment but the second or third to left. The rightmost neuron in the left bridge compartment does neither connect to the rightmost *glomeruli* in the *fan-shaped body*, but to the second or third to right. Instead, this is where the first of the CPU1b neurons of the right *protocerebral bridge* compartment arborizes. In this way, the vectorial directions encoded by the *protocerebral bridge* are rotated and wrapped by one angular resolution before being mapped to the corresponding directional *glomeruli* in the *fan-shaped body.*

Note that the *protocerebral bridge* is divided in the monarch butterfly (figure 5.5), while it forms one connected structure in *Megalopta genalis* (figures 5.1 and 5.2), *Scarabaeus satyrus* (figure 5.4), and most other insects (Heinze and Reppert, 2012). However, functionally it can be assumed equivalent to the two halves of a connected bridge.

The last involved neuron type is the *pontine* neurons which connect cells in the *fan-shaped body* across the two brain hemispheres. In the model, they receive excitatory input from CPU4 which they project inhibitorily to the CPU1 population associated with the opposite brain hemisphere in a way that rotates the encoded direction by 180°. Since the *pontine* neurons exert inhibition, the stored direction in CPU4 is effectively projected differentially to CPU1. I.e., the more active CPU4 is overall, the more pronounced is the transmitted activation difference between the most active and least active direction in CPU1. The sum over all direction encoding activities remains the same however, independent of the average population activity

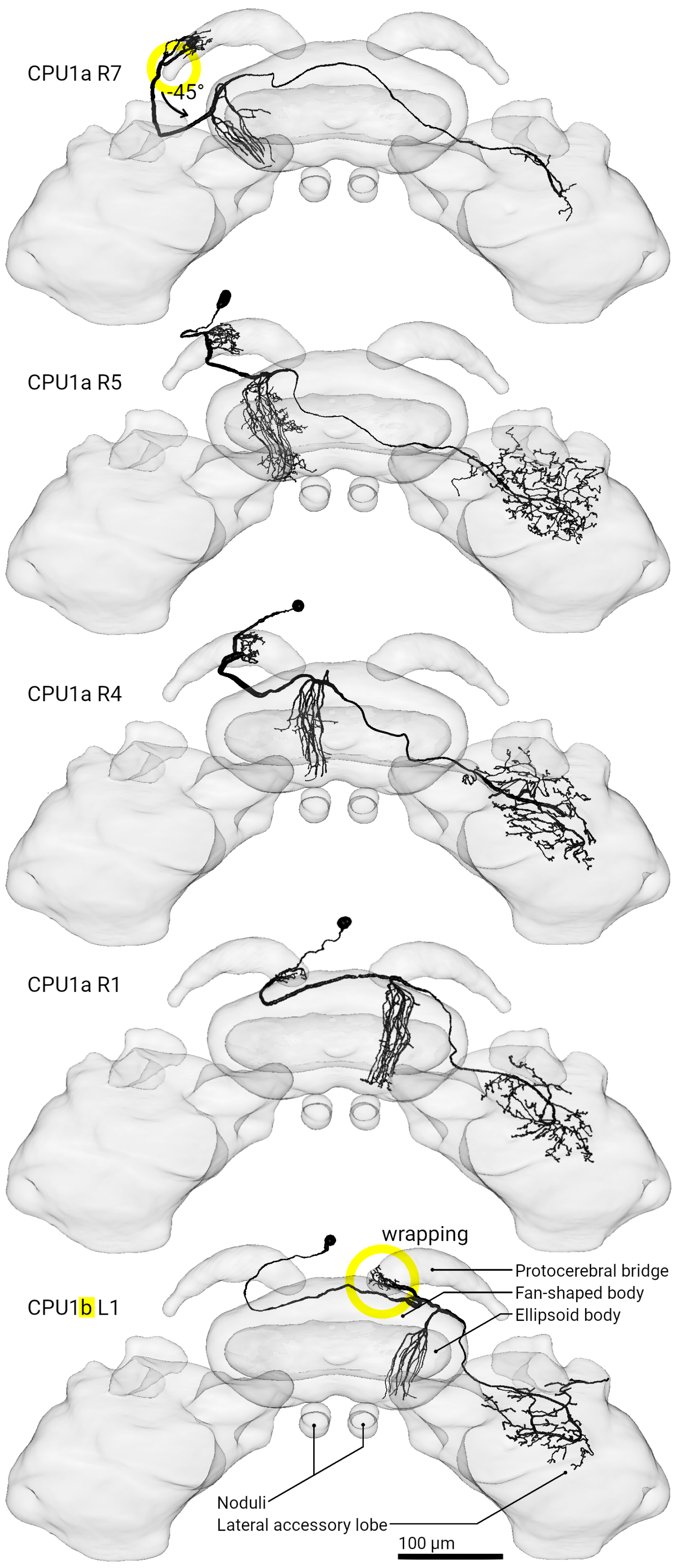


Figure 5.5: Various CPU1 neurons in the brain of the monarch butterfly *Danaus plexippus*. From top to bottom occurs a transition from left to right *glomeruli*. In the top figure it can be observed that CPU1aR7 does not connect the leftmost compartment of the *fan-shaped body* to the leftmost *glomerulus* in the *protocerebral bridge*, but instead to the second to the left. This effectively results in a phase shift of one angular resolution (-45°). In the lowest image one can see how a CPU1b neuron accomplishes the wrapping by taking the next orientation missing in the left *protocerebral bridge* compartment from the right one. The individual images are taken from insectbraindb.org based on data from Heinze and Reppert (2012).

of all CPU4 neurons. The summed activity of each lateral CPU1 population is then finally taken as the steering signal.

The concept of the entire encoding scheme becomes more obvious, when it is reduced to an abstract point neuron representation, which follows in the next section.

5.3.2 Abstract connectivity

Figure 5.6 shows the network in this representation. Based on the above mentioned observations, the neurons in TL, CL1, TB1, CPU4, and in the *pontine* population can each be assigned one of eight unique directions. The order of directions is preserved by most of the population interconnects, as they implement one-to-one mappings between the individual neurons. For example, the vectorial compass information from the 16 TL cells (eight per brain hemisphere) is translated through CL1 down to TB1. Note that, the TL input neurons inhibit the CL1 cells, which effectively inverts the activity after the first layer. However, this is without effect on the network function since only the orientation encoding is affected, not the orientation itself. As both of the CL1 neurons belonging to one directional component are branching into the same dendritic regions that the corresponding TB1 neuron has in the two *protocerebral bridge* compartments, the duplicated vectorial information collapses into a single one.

The inhibitory TB1 population then projects to the two lateral CPU4 and CPU1 populations, again preserving the vectorial alignment. CPU4 receives further input from the speed encoding TN neurons in a one-to-all mapping. That is, every CPU4 neuron belonging to one hemisphere receives the same excitatory hemispherical TN input. CPU4's output goes to the *pontine* populations in the same one-to-one way and to the CPU1 populations in a different way. Here, every CPU1 neuron collects excitatory input from one CPU4 neuron but cyclically permuted by one angular resolution relative to the TB1 input, clockwise, or counterclockwise, respectively. In that way, CPU1 compares the stored home vector from CPU4, rotated to the left or to the right, with the current heading direction.

To understand how this influences the overall activation of CPU1, let us consider that the insect currently heading into a direction that is off from the target direction by one angular resolution. In effect, in one lateral CPU1 population, the inhibitory TB1 activity will cancel out the activity that represents CPU4's stored home vector, as they exactly overlap. In the other lateral CPU1 population, though, some of the CPU4 activation will not be cancelled out and eventually causes some CPU1 neurons to respond stronger. The summed CPU1 activity in each hemisphere is therefore indicating whether the currently taken heading direction is deviating more to the left or to the right. If both lateral populations have the same average activity, the heading direction is parallel to the home vector. Thus, a steering signal can be derived from the two lateral CPU1 activities.

Lastly, CPU1 also receives inhibitory input from the *pontine* neurons that repeat the CPU4 activities but rotated by 180° and acting as inhibitors. As mentioned above, the effect is that the stored home vector is projected sort of differentially to CPU1. For example, if one CPU4 neuron has the same activity as the neuron

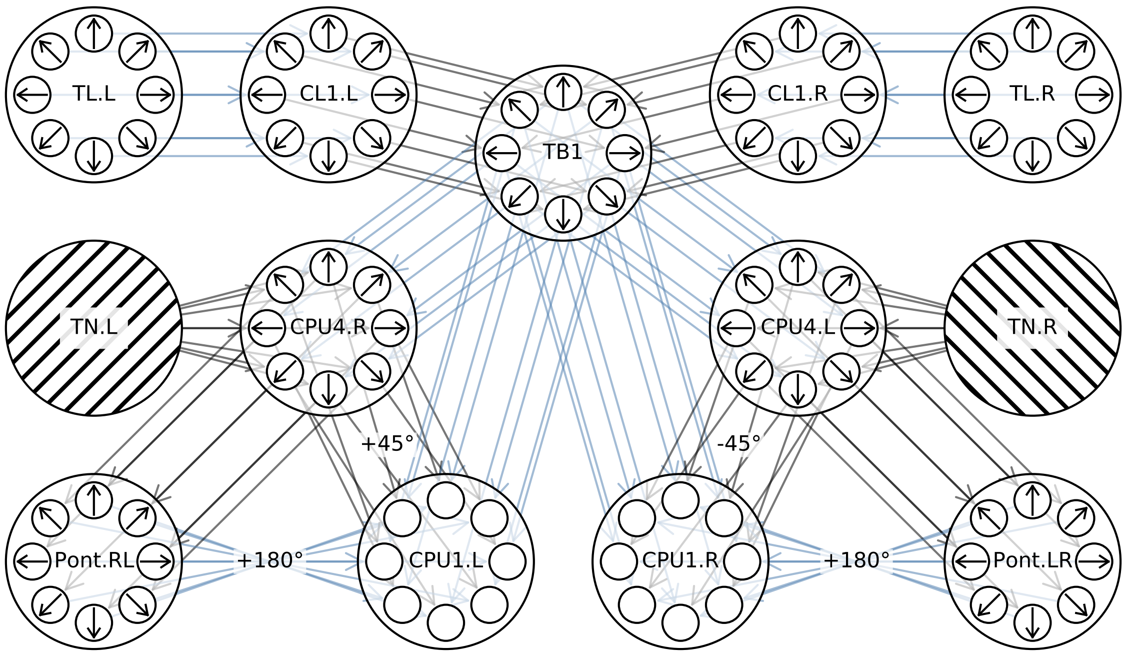


Figure 5.6: Abstract point neuron representation of the path integration network. Black denotes excitatory, blue inhibitory synapses. Except for TN, each subpopulation contains eight neurons. The connections from CPU4 to CPU1 are shifted by one angular resolution (i.e. 45°) clockwise or counterclockwise, respectively. The *pontine* populations project their activity inhibitorily to CPU1 rotated by 180°). All other connections preserve the orientational population encoding by implementing a one-to-one mapping of the individual neurons. The direction a neuron represents is indicated by the arrows within.

representing the opposite direction, their summed activities even out. Those neurons in CPU4, however, that indicate the home vector with a higher than average activity, face weak activations in the opposite direction, due to the integration rule of the CPU4 cells[1]. The high positive postsynaptic influence of active CPU4 cells is therefore matched by only weak inhibitory influence and vice versa. Thus, the postsynaptic activity pattern, representing the home direction over the receiving cells becomes more pronounced. Moreover, it becomes less dependent on the activity of CPU4 averaged over the population, since a higher overall activity also causes a higher overall inhibition, making the entire regulation mechanism more robust.

In Stone et al. (2017), all neurons are modeled as firing rate neurons with a sigmoidal transfer function

$$r_j = \frac{1}{1 + e^{-(a \cdot I_j - b)}} \tag{5.1}$$

of their accumulated synaptic input

$$I_j = \sum_i w_{ij} \cdot r_i. \tag{5.2}$$

[1]The rule is presented in detail in section 5.3.4

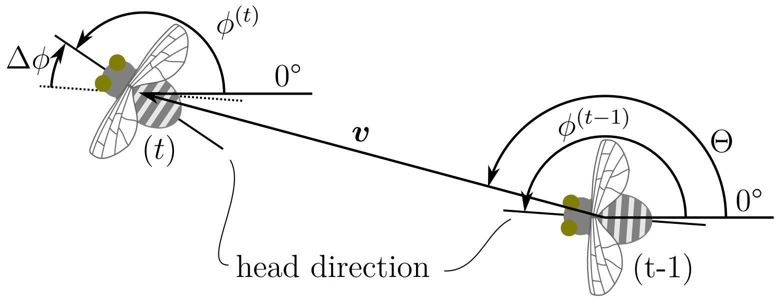


Figure 5.7: Coordinate system of the insectoid agent. Θ is the direction of flight and $\phi^{(t)}$ and $\phi^{(t-1)}$ the head direction of the current and the previous time step, respectively. $\Delta\phi$ is the difference between both and indicates the head rotation. As the time interval is discrete and defined to be 1, the traveled distance is equal to the velocity vector $\boldsymbol{v}$.

w_{ij} is the synaptic weight that connects neuron i with neuron j, and a and b are parameters that are individually tuned for every neuron population. The output rate is dimensionless and intrinsically normalized to the interval $r_j \in]0;1[$. While equation (5.1) is indeed universal for all neurons, the internal state of the cells is computed differently for the sensor populations TN and TL, and for the integrator population CPU4.

5.3.3 Sensors

The physical state of insect is described by five state variables, i.e., its position $\boldsymbol{x}$, its head orientation ϕ, and its velocity $\boldsymbol{v}$, where $\boldsymbol{x}$ and $\boldsymbol{v}$ are two-dimensional vectors. When switching to polar coordinates, there are two more derived state variable, the direction of movement Θ, and the absolute velocity v:

$$\boldsymbol{v} = \begin{bmatrix} v \cdot \cos(\Theta) \\ v \cdot \sin(\Theta) \end{bmatrix} \tag{5.3}$$

A graphical overview is given in figure 5.7. Note that the head direction does not necessarily have to be identical with the direction of movement. In this way, effects like side wind can be integrated into the model to study holonomic motion. Generally, a system is said to be holonomic when the number of degrees of freedom matches the number of controllable degrees of freedom, such that any system state can be achieved independently of the path taken. For example, human walking capabilities can be described as holonomic because an arbitrary potentially accessible state can be achieved independent of the path taken. A counterexample would be a wheel that rolls over a surface. For such a wheel positions and orientations are not independent of its rotation. Translated to the problem at hand, the insect's direction of movement is said to be nonholonomic when it depends on the head direction.

Optical flow

According to Stone et al. (2017), the internal states of the two TN cells are derived as

$$I_{\mathrm{TN},L/R} = \Big[\sin(\phi \pm \phi_{\mathrm{TN}}) \quad \cos(\phi \pm \phi_{\mathrm{TN}})\Big] \cdot \boldsymbol{v}. \tag{5.4}$$

Here, ϕ_{TN} is the "preference angle of a TN-neuron, i.e., the point of expansion of optic flow that evokes the biggest response" (Stone et al., 2017). For example, if $\Theta = \phi + \phi_{\mathrm{TN}}$, the response of the corresponding cell would be maximal. If, on the other hand, $\Theta = \phi + \phi_{\mathrm{TN}} \pm 90°$, the response of the corresponding cell would be minimal, since the agent moves orthogonally to the cell's sensitivity. In polar coordinates, equation (5.4) can be written as

$$\begin{aligned} I_{\mathrm{TN},L/R} &= v \cdot [\cos(\Theta)\sin(\phi \pm \phi_{\mathrm{TN}}) + \sin(\Theta)\cos(\phi \pm \phi_{\mathrm{TN}})] \qquad (5.5)\\ &= v \cdot \sin(\Theta + \phi \pm \phi_{\mathrm{TN}}). \qquad (5.6) \end{aligned}$$

Since the cell response is dependent on $\Theta + \phi$ rather than on $\Theta - \phi$, $I_{\mathrm{TN},L/R}$ is not rotationally invariant. The author of this thesis supposes, however, that $I_{\mathrm{TN},L/R}$ should be invariant against common changes in Θ and ϕ, because the optical flow should only depend on the insect's frame of reference. E.g., if the insects heads north ($\Theta = 90°$) with its head tilted by 20° relative to its direction of movement ($\phi = 20° + \Theta = 110°$), it should measure the same optical flow as if it was heading eastwards, that is $\Theta = 0°$, with the same relative head tilt. As the TN responses, depicted in Stone et al. (2017) are also supporting this assumption, the author of the thesis at hand suspected a minor mistake in the written equations and therefore derived an own solution based on an orthogonal projection of $\boldsymbol{v}$ onto a plane that reflects the preferential direction of each TN neuron:

$$P_{\mathrm{TN},L/R} = v \cdot \sin(\Theta - \phi \pm \phi_{\mathrm{TN}}) \tag{5.7}$$

This, however, only accounts for translational movements. If the insect's rotation is further included, an additional term emerges:

$$P_{\mathrm{TN},L/R} = v \cdot \sin(\Theta - \phi \pm \phi_{\mathrm{TN}}) \mp \rho \cdot \dot{\phi} \tag{5.8}$$

Here, the constant ρ is proportional to the distance between the two eyes. Because the optical flow should be positive for both cells when the insect is moving forward into head direction, the sign of the translation term of one eye has to be inverted. Finally, the cell state of the TN cells is obtained as

$$I_{\mathrm{TN},L/R} = \pm v \cdot \sin(\Theta - \phi \pm \phi_{\mathrm{TN}}) \mp \rho \cdot \dot{\phi}. \tag{5.9}$$

In Stone et al. (2017), the term $\pm\rho \cdot \dot{\phi}$ is completely omitted, although TN signals that are plotted of over time suggest otherwise.

At this point, it is also worth mentioning that although Stone et al. elaborate on holonomic movements, the case where Θ is indeed independent of ϕ is only tackled in the supplementary material of the publication and without a comprehensive functional description. In the main body of the paper, they only investigate "partial holonomic motion" by which they mean a constant non-zero offset between Θ and ϕ. This, however, is actually unrelated to the definition of holonomic, because Θ and ϕ remain fully dependent on each other. Therefore, this "partial holonomic" case should not be regarded as holonomic but can be interpreted as a static rotation of the flow field organs with respect to the body. In the implementation at hand, the insect is constrained to always move into its head direction instead, i.e., $\Theta = \phi$. Hence, equation (5.9) simplifies to

$$I_{\mathrm{TN},L/R} = v \cdot \sin(\phi_{\mathrm{TN}}) \mp \rho \cdot \dot{\phi}. \tag{5.10}$$

Head direction

The second class of sensor signals comes from the TL compass neurons. This population comprises two groups of eight neurons TL_j with $j = \{0, .., 7\}$ that encode the head orientation ϕ in their output activity:

$$I_{\mathrm{TL},L/R,j} = \sin\left(\phi + \frac{j \cdot \pi}{4}\right). \tag{5.11}$$

After being warped by the transfer functions (equation (5.1)), the signal is then passed on to the 16 CL1 neurons and further to the eight TB1 neurons. The latter also experience lateral inhibition from their neighbors such that their internal state becomes

$$I_{\mathrm{TB1},j} = \frac{2}{3} r^{(t)}_{\mathrm{CL1},j} + \frac{1}{3} \sum_{i=0}^{7} w_{ij} r^{(t-1)}_{\mathrm{TB1},i}. \tag{5.12}$$

Here and in the following, t denotes the discrete time. Moreover, there is no self inhibition, i.e., $w_{ij} = 0$, if $i = j$. The lateral inhibition to all other cells in the population functionally implements a winner-take-all, or ring-attractor network. Among the population, this causes weakly activated cells to become even weaker due to the inhibition of their neighbors, while highly active cells receive only weak inhibition from their more inactive neighbors. In effect, TB1 also represents the compass information, only weakly temporally filtered and morphed in amplitude by the lateral inhibition and the transfer functions.

Because of the low number of available neurons on the BrainScaleS-2 prototype chip, the directional sensor input is therefore directly provided to the network from the TB1 cells, omitting TL and CL1. A detailed summary of the encoding follows in section 5.4.

5.3.4 Integrators

As indicated above, the CPU4 integrators form another exception to equation (5.2). Their internal cell state

$$I^{(t)}_{\mathrm{CPU4},L/R,i} = I^{(t-1)}_{\mathrm{CPU4},L/R,i} + h \cdot \left(r^{(t)}_{\mathrm{TN},L/R} - r^{(t)}_{\mathrm{TB1},i} - k\right) \tag{5.13}$$

depends on the inputs from TB1 and TN, as well as on their own previous state. The two constants h and k define the coupling strength and the damping, respectively.

5.3.5 Motoric outputs

Motoric output is derived from the activities of the left and right CPU1 populations. While the absolute velocity v is kept constant throughout an experiment, the direction is altered according to the difference of the summed lateral activities:

$$\Delta\phi = \Delta\Theta = \mu \cdot \left(\sum_i r_{\mathrm{CPU1},R,i} - \sum_i r_{\mathrm{CPU1},L,i} \right) \tag{5.14}$$

Here, μ is a heuristically chosen scaling constant.

This concludes the functional description for the model proposed in Stone et al. (2017). Next, the changes and adaptations that had to be introduced for the hardware implementation are introduced.

5.4 LIF-based model

5.4.1 From spike rates to spikes

In section 2.1.2, in particular figure 2.9, the transition of the LIF equations to the stochastic rate limit has already been qualitatively introduced and illustrated. The equations (5.1) and (5.2) that are used to describe the rate response of the TB1, CL1, CPU1 and the *pontine* populations present a commonly used quantitative approach to approximate the rate output of spiking neuron models under stochastic influence (Stevens and Zador, 1996; Dayan and Abbott, 2001; Fourcaud and Brunel, 2002). The implementation of these neuron populations on the BrainScaleS-2 prototype is therefore straight-forward.

However, since LIF neurons reset their internal state after each firing event (see equations (2.1), (2.3) and (2.4)), the integration dynamics of equation (5.13) cannot be implemented as easily. As the path integration time constants, i.e., multiple seconds or minutes in biological time, by far exceed the time scales of the LIF dynamics, i.e., 1 ms to 10 ms, only an extraneuronal mechanism or a large population of LIF neurons can account for the homing memory mechanism. One example for the latter are synfire chains (Abeles, 2009). These chains describe concatenated feed-forward networks that are characterized by sequential layer-wise spiking activities that propagate through the network in a wave-like manner. The time it takes for such a wave to traverse the entire synfire chain is proportional to the time constant of the neurons multiplied by the number of layers. Since there are only about 18 CPU4 neurons per *glomerulus* in the insect brain (Stone et al., 2017), even a synfire chain with only one neuron per layer, could only account for a wave period length of less then 100 ms. The idea of encoding of the integrated distance as a population activity within one CPU4 *glomerulus* is therefore omitted.

Instead, an extraneural mechanism is considered. One such mechanism that operates on suitable time scales is axoaxonic synaptic modulation.

This mechanism plays a prominent role in the gill and siphon withdrawal reflex in the sea snail *Aplysia* (Kandel, 1976) and is commonly found in insect brains (Schneider-Mizell et al., 2016) too. Usually, these synapses are formed between

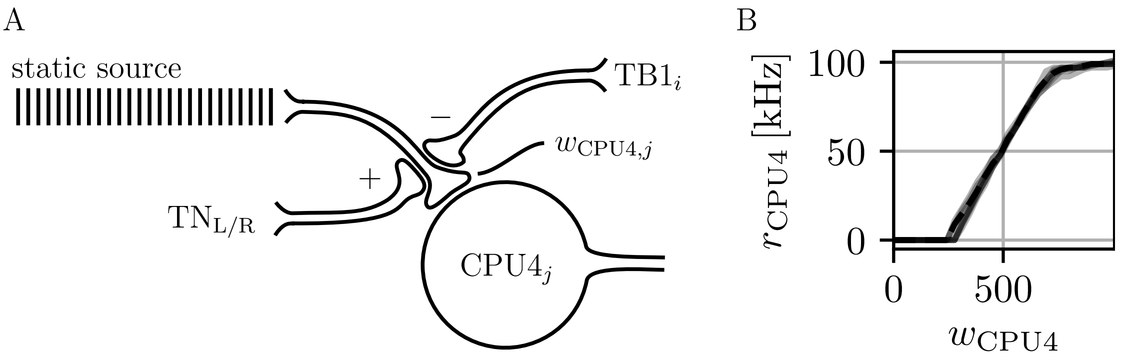


Figure 5.8: Axoaxonic depression and facilitation. A) A synapse that forms an excitatory connection to a background source with a constant spike rate is modulated by two axoaxonic synapses. Activity coming from the TN$_{\mathrm{L/R}}$ axons facilitates the synaptic weight $w_{\mathrm{CPU4},j}$, while activity from TB1$_{\mathrm{i}}$ inhibits it. The neuron responds with an output rate $r_{\mathrm{CPU4},j}$. B) Output rate vs. weight. The plot shows an overlay of the transfer functions of eight CPU4 neurons on the BrainScaleS-2 system.

facilitating neurons and the presynaptic region of a normal axodendritic synpase (Kandel et al., 2013). The strength of this synapse is then reduced or increased on short-term memory time scales by the activity of the modulation neuron. The mechanism is also known as presynaptic facilitation or depression (Byrne and Kandel, 1996; Lodish et al., 2000). In the case of the modelled CPU4 cells, these neurons correspond to the TN and TB1 subpopulations. In order to reproduce the behavior of equation (5.13), one spike source with a constant average rate is connected to each integrating neuron by an excitatory synapse with a variable weight $w_{\mathrm{CPU4},j}$. The integrating neuron therefore responds with an average output spike rate $r_{\mathrm{CPU4},j}$ that is proportional to this weight. To that end, the synapse can operate on the short-term memory time scales that are necessary for the distance integration. Thus, in analogy to equation (5.13), we have

$$w^{(t)}_{\mathrm{CPU4},L/R,i} = w^{(t-1)}_{\mathrm{CPU4},L/R,i} + \frac{h}{r_{\max}} \cdot \left(r^{(t)}_{\mathrm{TN},L/R} - r^{(t)}_{\mathrm{TB1},i} - k \right) \tag{5.15}$$

Figure 5.8 provides a schematic illustration of the axoaxonic synapses and the measured response on the BrainScaleS-2 prototype setup (more details follow in section 5.5.2).

With the implementation strategies laid out, the neuronal properties and details can now be discussed.

5.4.2 General neuron properties

Unlike the transfer function of the fire rate neurons from Stone et al. (2017) (see equation (5.1)), the LIF equations do not intrinsically limit the output rate to a dimensionless interval $]0;1[$. Instead, they produce a series of spikes, whose rate response with respect to the input is determined by the equation parameters, most importantly, the leak potential V_{leak}, the threshold potential V_{th}, the reset potential V_{reset}, the membrane time constant τ_{mem}, and the refractory period τ_{ref} (see section 2.1.2). Within certain limitations that the hardware implementation imposes, these parameters can be freely configured and calibrated to obtain a transfer function that best matches the desired biological blueprint. The minimum output rate is, of course, naturally chosen to be $0\,\mathrm{s}^{-1}$, i.e., no output activity at all.

On the other hand, a maximum sustained rate of 150 Hz has been recorded in the TN cells in *Megalopta genalis* (Stone et al., 2017). Although this rate is well within reach for the hardware LIF neurons, the implemented upper limit is chosen slightly lower to 100 kHz, which is equivalent to a biological rate of 100 Hz. The main reasons for this reduction are the tight timing constraints of the code execution on the PPU. The desired maximum output spike rate also determines the rate at which the spike sending routine that implements the background generators for the CPU4 weight memories and the sensor signals has to be triggered. The more often this routine is called, the less time the PPU has left in between the individual calls for other tasks, like the simulation update or data recording.

However, a deviation by −33 % with respect to the biological maximum rate is in fact largely negligible, especially since the 1000 fold acceleration of the hardware is to be understood as approximative as the typical time scales of the neuronal phenomena. As outlined in section 2.1.2, 100 Hz is generally considered a high firing rate that also corresponds to typically measured maximum rates of cortical neurons.

Therefore, the output range of the BrainScaleS-2 neurons is confined to 0 kHz to 100 kHz. This rate spectrum applies to all involved neurons as well as to all sensors. Obtaining this range requires a few calibration schemes that are described in the following.

5.5 Insects on BrainScaleS-2

Generally, the calibration becomes necessary for two reasons. First, parameters like, for example, the membrane time constant cannot be set directly in SI units, but may depend even nonlinearly on the corresponding DAC values of the capacitive memory (see section 3.1.4). Second, fixed pattern noise due to unavoidable chip manufacturing variations causes mismatch between the individual neurons and synapses. Therefore an adequate calibration scheme has to be employed first in order to obtain a reliable and uniform set of neurons.

5.5.1 Non-integrating neurons

The goal for this neuron type is to respond to a superposition of excitatory and inhibitory inputs in a uniform and reliable way that also respects the desired output range. Before the precise calibration procedure starts, all neuron are coarsely calibrated by a database-based configuration routine (Aamir et al., 2018). These routines have been developed by Yannik Stradmann in the Python language to mitigate fixed-pattern noise on the 2nd BrainScaleS-2 prototype chip. Instead of setting the capacitive memory parameters manually, they allow the user to describe the desired neuron properties in SI units. For example, if the membrane time constant is set to 3 µs, the capacitive memory cell that defines the leak resistance will automatically be set accordingly. With this procedure the neurons are initially put into heuristically chosen configuration that represented a good basis for the succeeding rate calibration.

$$\begin{aligned} \tau_{\text{syn,inh}} &= 1.0\,\mu\text{s} \qquad & V_{\text{th}} &= 1.0\,\text{V} \\ \tau_{\text{mem}} &= 1.5\,\mu\text{s} \qquad & V_{\text{reset}} &= 0.0\,\text{V} \\ \tau_{\text{ref}} &= 2.0\,\mu\text{s} \end{aligned}$$

While the time constants lie in ordinary ranges, the ratio $\frac{V_{\text{th}}-V_{\text{leak}}}{V_{\text{leak}}-V_{\text{reset}}} = 19$ is atypical for biological neurons. The reason for this is that V_{leak} is one of the control variables of the second calibration routine that is responsible for adjusting the actual rate response characteristics. Setting V_{leak} initially to 50 mV guarantees that the control variables converge all from the same polarity, thus, stabilizing the calibration procedure.

Moreover, V_{reset} is usually set to values higher than 0.0 V, because a lower dynamic range of the membrane voltage decreases nonlinear distortions, which is important if the LIF equations ought to be replicated faithfully. Here however, maximizing the dynamic range of the neuron membrane was found to be beneficial for smooth response functions.

The main calibration routine is then executed on the PPU and comprised of three consecutive bisections. The first one targets for a 50 kHz firing rate at 50 kHz excitatory input with 0 Hz inhibitory input and uses V_{leak} as the control variable. The second bisection tunes the refractory period by setting the corresponding bias current `i_refr` to achieve 100 kHz output rate at 100 kHz excitatory input and again without inhibitory input. Finally, the inhibitory synaptic time constant is tuned with the corresponding bias current to such that 50 kHz output rate result from 100 kHz excitatory input and 100 kHz inhibitory input.

The result of this calibration scheme is depicted for eight CPU1 neurons in figures 5.9 and 5.10.

The target response that is used to derive the deviation in figure 5.10 is naively expected to be

$$r_{\text{out}} = r_{\text{exc}} \cdot (1 - c \cdot \frac{r_{\text{inh}}}{r_{\text{max}}}). \tag{5.16}$$

$c = 0.5$ is set such that the 50 % output rate is achieve when both inputs fire at 100 %. Note that equation (5.16) does not represent a sigmoidal transfer function as equation (5.1) suggests. Instead, the output rate increases linearly with r_{exc} and decreases linearly with r_{inh}. However, since the CPU1 neurons' output activity only affects the insect's behavior in terms of the difference between the summed activities of the left and right subpopulation, this has no effect on the network function. Moreover, the output response is most crucial at the operating point of the neurons, that is in the middle of the output range. Here, the desired behavior is achieved as the corresponding activities of the individual neurons are well aligned at the correct voltage.

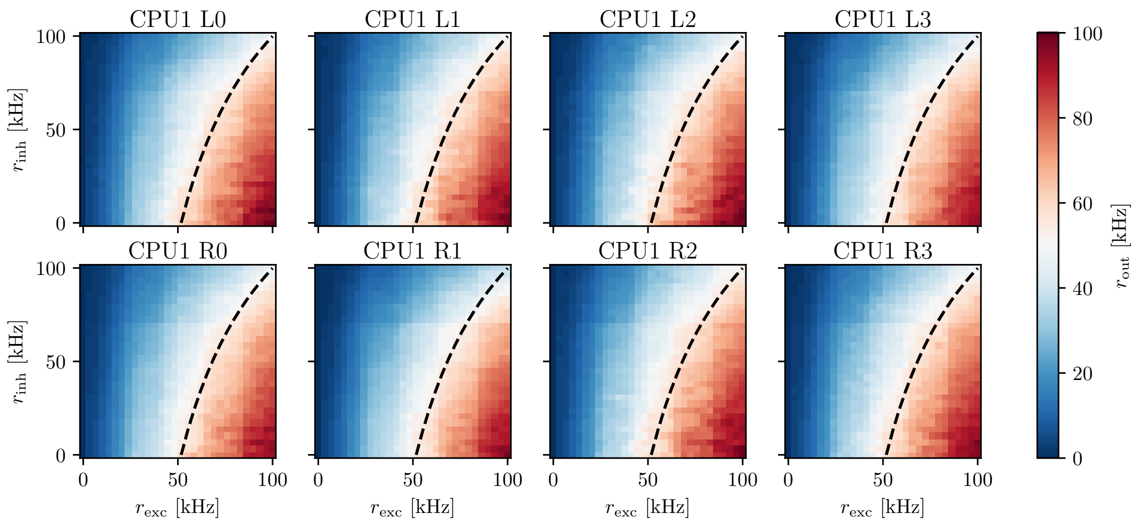


Figure 5.9: Output response of the calibrated CPU1 neurons with respect to excitatory and inhibitory input rates. The dashed line indicates the region where 50 % of the maximum output rate is expected.

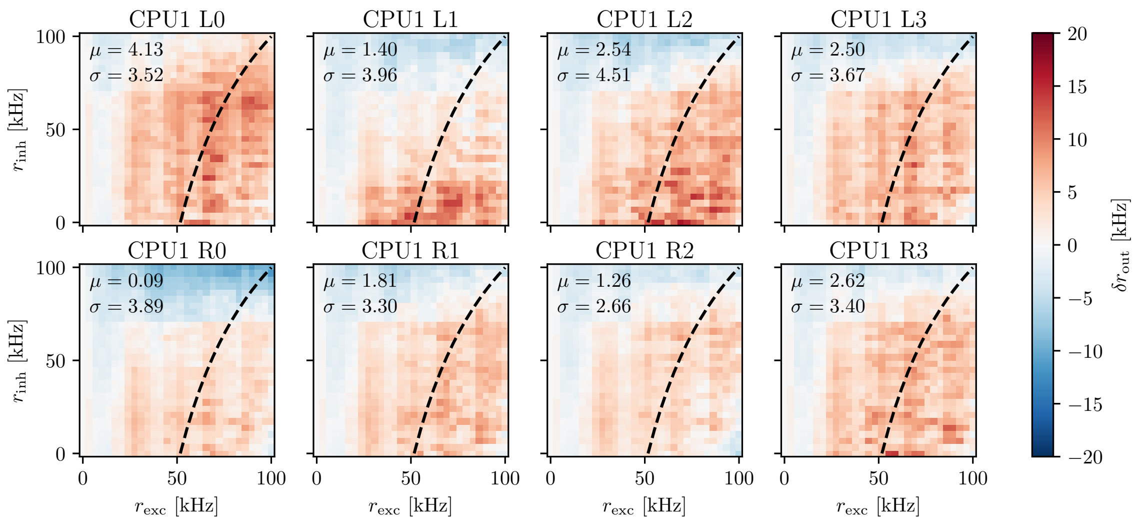


Figure 5.10: Deviation of the output responses of the calibrated CPU1 neurons from an ideal response behavior. The dashed line indicates the region where a 50 % output rate is expected. μ is the mean deviation from the expected response in kHz and σ the standard deviation of the differences. The deviation pattern is similar for all eight neurons, such that the inter-neuron deviation is less significant than the deviation from the target distribution.

5.5.2 Integrator neurons

The calibration of the CPU4 neurons also starts with a database-based precalibration that sets the neurons initially to the same parameter set as in section 5.5.1. The main calibration is divided into two bisections, that again calibrate the response at the operating point $r_{\text{out}} = 50\,\% \cdot r_{\text{out}}$ and the maximum rate. However, as mentioned earlier, the CPU4 neurons function fundamentally different from the other neurons as their response depends on the axoaxonic synaptic weight rather than on an input spike rate. Since the hardware synapses on BrainScaleS-2 have discrete weights with a resolution of six bit, the dynamic range of a single synapse is limited to values between 0 and 63. As the home vector is basically stored in these synapses, their dynamic ranges effectively determine the ratio between the accessible spatial range and the precision of the stored home location. It is therefore vital to extend this dynamic range if possible.

In order to do so, 16 synapses were joined together into a supersynapse. Technically, this is achieved by combining 16 synapses per CPU4 neuron vertically in the top 16 rows of the synapse array. These synapses are set to the same address such that all of them receive static background activity as it is required for the axoaxonic integrators. The synaptic weights are then filled from the top to the bottom. The following pseudo code illustrates the procedure.

```
weight = 543

for row in [0 .. 15]:
    if weight > 63:
        synapse_weight[row] = 63
        weight -= 63
    else:
        synapse_weight[row] = weight
        weight = 0
```

Thus, the value range per supersynapse is extended to $w_{CPU4} \in [0; 63 \cdot 16] = [0; 1008]$, corresponding to almost 10 bit.

However, by default the global bias current that defines the amplitude of the synaptic current pulses is tuned for optimal single-synpsase response. Therefore, the current injected by 1 to 16 synapses simultaneously is too high and enough to saturate the synaptic input line. The synaptic bias current was therefore reduced to allow a decent response of the 16 joint synapses.

The background input to the CPU4 supersynapses is a periodic spike train with static spike rate of 100 kHz. Again, this fixed-rate background is implemented due to the tight timing demands of the code execution on the BrainScaleS-2 prototype. Spike sources with Poisson-distributed inter-spike intervals would be favorable because they would reduce aliasing artifacts, soften discontiuous transistions in the output response, and be biologically more plausible. However, in simulation it was found that the overall performance is not dramatically affected by the fixed rate sources. Moreover, in hardware, jitter and intrinsic analog noise additionally mitigate possible artifacts to some degree.

The control variables for the two calibration bisections are the source degeneration bias of the excitatory synaptic input operational transconductance amplifier for

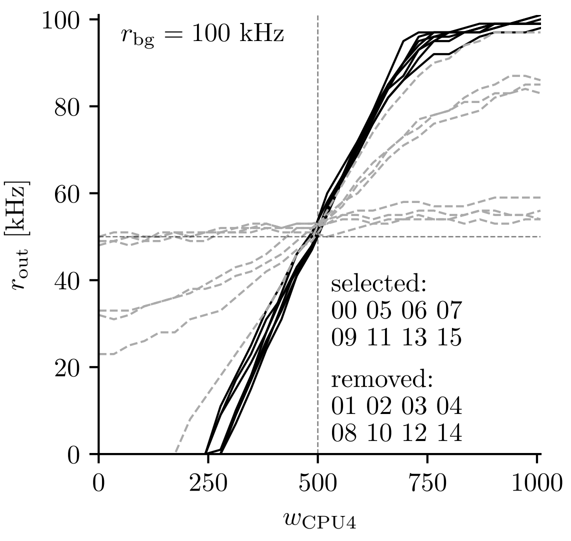


Figure 5.11: Output response of the CPU4 neurons with respect to the weight of the axoaxonic supersynapses. The dark solid curves represent eight selected, and the light dashed curves the neurons which are sorted out due to bad response characteristics. The neuron IDs are listed in the lower right quadrant.

$r_{out} = 50\,$kHz and again the bias current that determines the refractory period for the maximum output rate.

While the calibration generally succeeds in producing a sufficiently large set of close-to-optimal CPU4 neurons, it is not capable of configuring all neurons into the desired operating mode. This is mostly due to two aspects. First, neither the synapses nor the synaptic input were designed to operate in the supersynapse regime. Although the system is perfectly capable of handling the corresponding parameter settings, it has not been designed for that purpose. Therefore, some of the necessary parameter settings are close to the edge of the allowed ranges, where nonlinearities and clipping effects start to occur. Second, the relatively low periodic input spike rate causes aliasing artifacts that can confuse the calibration routines. As already mentioned, a Poisson input spike train at a higher rate would virtually remove these artifacts and allow for a much more stable calibration procedure and output response. However, the mis-calibrated neurons are eventually separated from the functioning ones and used as CPU4 cells. Figure 5.11 shows the response curves of 16 neurons after calibration and indicates which ones are either sorted out or employed in the network.

The obtained response functions match well in slope and offset and therefore suited for implementing the memory mechanism. It is important to note that the experiment starts with all CPU4 weights set to 504. If the network is configured correctly, it will steer the virtual insect such that the activity pattern among CPU4 equalizes and this initial weight configuration is approximately restored. It is therefore important that the neuron activities match well around this operating range but it is less significant if they deviate a at the edges. Moreover, a steep slope around $w_{CPU4} \approx 500$ is therefore also beneficial as it amplifies the network response with respect to changes in the synaptic weights that reliably store the homing information. The calibrated neurons fulfill these criteria superbly.

5.5.3 CPU4 weight access

Since the CPU4 weights have to be accessed frequently to implement the axoaxonic modulations in parallel to the continuously running hardware neurons, the corresponding software routines on the PPU have to be as efficient as possible. Therefore, the supersynapse access functions make excessive use of the assembly instructions of the PPU's vector unit. As it would shift the focus too far from the actual substance of this chapter, the read and write routines are not given below but are attached as commented code snippets in the appendix. The read function can be found in appendix B.1 and the write function in appendix B.2.

One aspect that is important to mention here, however, is that the vector unit always operates on slices of 16 columns at once. Therefore, the optimized read and write functions affect all synapses of the first 16 neurons, irrespective of whether some of these neurons are actually used as CPU4 integrators or not[2]. For that reason, the upper 16 synapse columns for the first 16 neurons are not available for other purposes.

5.5.4 Reduced network size

Since the resources are limited to maximally 32 available neurons on the BrainScaleS-2 prototype chip, the original network size from Stone et al. (2017), i.e., 90 neurons, is too large and therefore had to be scaled down.

As explained in section 5.3.3, the first step to reduce this number is to inject the compass information directly from TB1, omitting TL and CL1.

Moreover, as the neuromorphic system is not subject to Dale's law (Strata and Harvey, 1999), the inverting *pontine* population can be left out as well, since CPU4 is able to project both excitatory and inhibitory input to CPU1 in an arbitrary configuration.

Since TN and TB1 are virtually implemented spike sources, both do not require physical neurons to exist. Consequently, 32 neurons remain, which just fit into the BrainScaleS-2 prototype.

However, during the experiment preparations hardware-independent software simulations of the network have been carried out as well. In the course of this, it turned out that the number of angular cells per population does not have to be constrained to exactly eight. In fact, the network remained fully functional with only three neurons per subpopulation. For that reason, it was decided to cut down the angular resolution from eight to four, thus, reducing the number of neurons further down to only 16 cells, i.e., eight for the CPU4 integrators and eight for the CPU1 steering cells, divided into two hemispherical subpopulations, each. The remaining 16 neurons are employed for two other purposes.

First, four neurons mirror the TB1 sources, allowing the associated spikes to be recorded on the FPGA memory on the BrainScaleS-2 test setup.

[2]This a particular circumstance on the 2nd generation prototype, where not all vector instructions supported masking yet. On later chip versions, there is an option to apply vector operations only to a masked subset of entries within one slice.

Second, instead of summing over the two CPU1 subpopulations, two motor (M) neurons collect the CPU1 spike output from each hemisphere. Their calibration is equivalent to the CPU1 calibration, only that the synaptic input weight is reduced because it receives signals from four instead of one excitatory presynaptic partners.

Thus, 18 neurons implement the entire network, and 4 further neurons monitor the TB1 activities.

5.5.5 Sensor inputs

Adapted to only four instead of eight angular neurons, the sensor response of the four TB1 neurons can be obtained in analogy to equation (5.11) as

$$r_{\mathrm{TB1}_j} = \frac{r_{\max}}{2} \cdot \left[1 + \sin\left(\phi + \frac{j \cdot \pi}{2}\right)\right]. \tag{5.17}$$

This causes the virtual TB1 neurons to encode the head direction as an activity distribution over the four cells.

The odomotric TN input is derived in analogy to equation (5.10) as

$$r_{\mathrm{TN},L/R} = r_{\max} \cdot \left(\frac{v}{v_{\max}} \cdot \sin(\phi_{\mathrm{TN}}) \mp \rho \cdot \dot{\phi}\right) \tag{5.18}$$

with $\phi_{\mathrm{TN}} = \pi/2$ and $\rho = 0.614$. Setting $\phi_{\mathrm{TN}} = \pi/2$ eliminates the second factor in the first term. In the non-holonomic case, choosing this arbitrarily is allowed since it only corresponds to a linear scaling of the velocity contribution. Furthermore, $\frac{v}{v_{\max}} = 0.5$ as the velocity remains constant over the entire experimental run.

5.5.6 Motor output

The steering signal is derived from the activities of the two motor neurons as

$$\Delta\phi = \Delta\Theta = \frac{\mu}{r_{\max}} \cdot (r_{\mathrm{M},L} - r_{\mathrm{M},R}) \tag{5.19}$$

with $\mu = 0.982$. The velocity is again set to a constant value.

The author would like to add that both ρ from equation (5.18) and μ from equation (5.19) are actually represented as integer numbers in the PPU code (200 and 160, respectively). The reason for their fractional appearance here is that all angles on the hardware are not represented in radians but run from 0 to 1023. The conversion to radians is responsible for these inconveniences.

A summary of the entire implemented network is given in figure 5.12.

5.5.7 Experiment execution and scheduling

Each experimental run starts with transferring a set of parameters from the host computer to the BrainScaleS-2 system. This includes a random seed for the outbound journey, the experiment run time t_{stop}, the time at which the agent is to return t_{return}, the CPU4 decay k, the CPU4 update scaling h, and the synaptic weights. Subsequently, the code execution is started on the PPU.

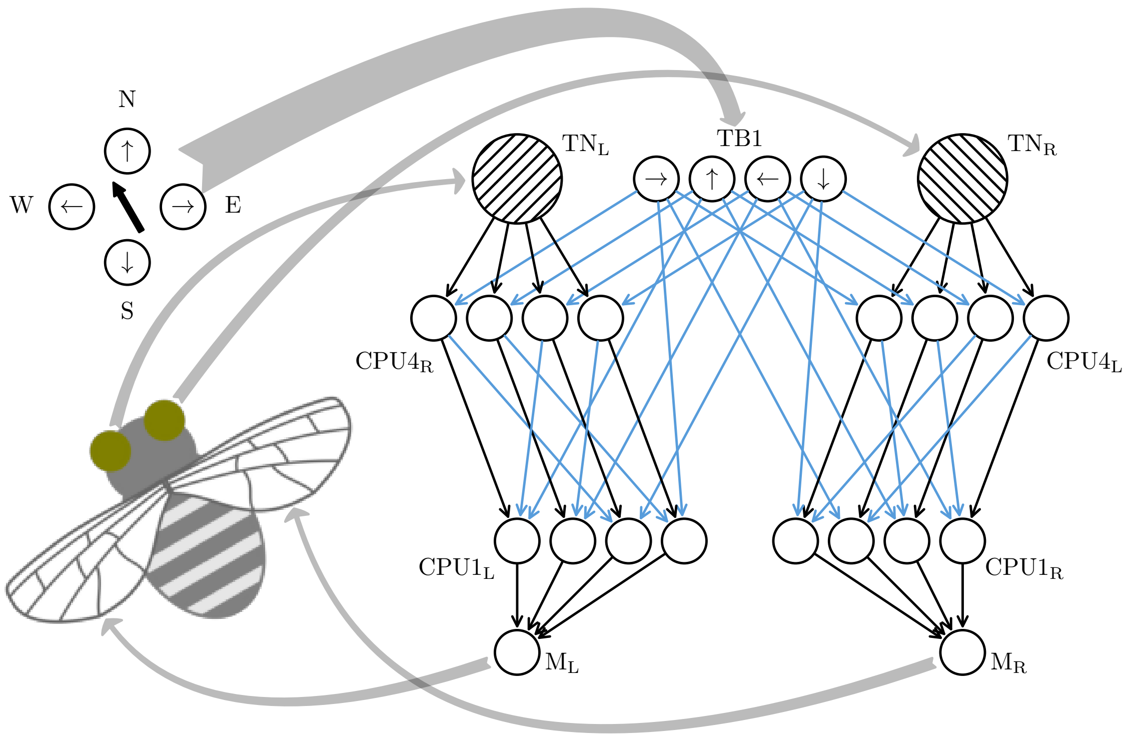


Figure 5.12: Network architecture of the BrainScaleS-2 implementation. The black and blue arrows correspond to excitatory and inhibitory connections, respectively. The first layer (TN and TB1) represents the sensor cells and is fully virtual. The CPU1 and CPU4 populations are divided into a left and right subpopulation, respectively.

During runtime, the PPU executes the simulation and implements all sensory input spike sources. One agent/environment update is conducted every $\Delta t = 100\,\mu\text{s}$, which corresponds to a frame rate of 10 Hz in biological time. To achieve the maximum spike rate of 100 kHz, the spike sources have to be able to send a spike every 10 µs. As the code execution time for one full agent/environment update exceeds the time between two consecutive spikes, it is divided into three subcycles that are called in between individual spike sending routines:

1. Iterate the insect state and transmit the sensory input to the TN and TB1 spike generators.
2. Read the CPU4 weights and calculate their updates.
3. Update the CPU4 weights, process the motor neuron output and update the agent velocity accordingly. Also update the simulation state, and store the trajectory data.

From the experiment start until $t = t_{\text{return}} = 50\,\text{ms}$, the virtual insect is forced on a random walk. This happens by overriding the motor neuron outputs by artificially generated pseudo-random signals. During this phase, the network does not influence the insect's motion but is provided with motion-derived sensory input. The velocity is set to the same value as in the brain driven mode and the steering signals are scaled to produce similar turning radii. At $t = t_{\text{return}}$, the agent switches from random to network-driven movement. Translated to biological time, the insect therefore spreads out for 50 s and has then 150 s to return to its nest. Finally, the simulation stops at $t = t_{\text{stop}} = 200\,\text{ms}$. At this point, the trajectory data and spikes are read back from the host computer.

The BrainScaleS-2 prototype chip can record all post-synaptic spikes using the connected FPGA, while the available on-chip memory for arbitrary data storage is limited to 4 kB. The latter is used for saving the trajectory from which the velocity and the optical flow can be reconstructed. Both the x and y coordinates are 16 bit signed integer values. Therefore, a total of 1000 locations can be stored. With $n = t_{\text{stop}}/\Delta t = 200\,\text{ms}/100\,\mu\text{s} = 2000$ time steps in one run, every second step is written to memory. Figure 5.13 gives an overview of the scheduling.

5.6 Execution and parameter sweeps

Figure 5.14 shows the network activity of an entire experimental run together with the trajectory. The progress is indicated at four different times. Before t_{return}, the insect performs a forced random walk while the home vector is stored in the left and right CPU4 populations. While the activity is approximately equal among these neurons at $t = 0$, it diversifies until the furthest distance from the home location is reached. The amplitude of this activity differentiation encodes the distance while the direction is encoded in the population.

After t_{return}, the motor neurons take over the agent movement and steer the insect back home, equilibrating the CPU4 spike pattern again. At $t \approx 100\,\text{ms}$ homing is completed and the insect starts looping around the home location.

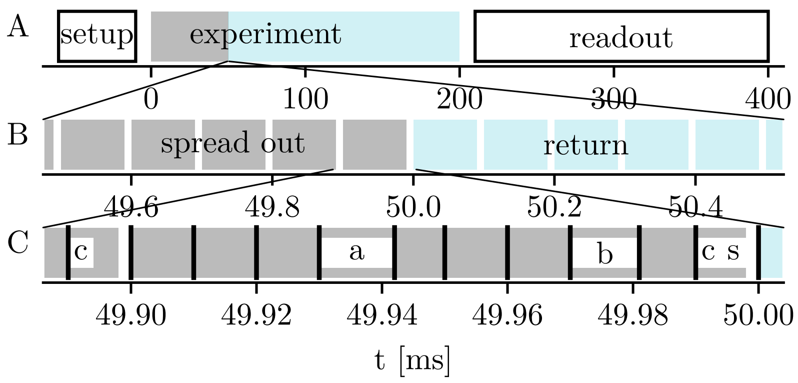


Figure 5.13: Experiment schedule. A) The experiment starts with a setup phase in which the parameters are transferred to the BrainScaleS-2 prototype and the synaptic weights are initialized. The actual experiment, starting with a spread out phase follows. After the full experiment, data is read back from the system memory to the host. The time required for readback varies, as it depends on the amount of spikes produced in an experiment. B) Zoom into the point of return. Each block symbolizes an agent/environment update, the blue color indicates the return phase. C) Zoom into one update. The black bars mark the times at which the spike sending routine is called. The three update phases are represented by the white blocks *1*, *2*, and *3*. Position recording happens only in every second update cycle in block *s*. Note that the variable time of the update phase executions can introduce a jitter in the following spike.

5.6.1 Parameter sweep

The influence if the parameters h and k from equation (5.15) on the dynamical behavior of integrating synapses is illustrated in figures 5.15 and 5.16.

The scaling value h determines the sensitivity of the integrator. If it is set too low, then the precision of the memorized home location decreases because every step causes only a small increment. This effect is visible in the lower left plot of figure 5.15 that shows the standard deviation σ of the trajectory during the looping phase. σ increases by tendency as h decreases. On the other hand, increasing h also increases the precision of the home location but also the risk, that the synaptic weights clip, which then distorts stored home location. This effect can be seen in the lower right plots of figure 5.15 that illustrates the standard deviation of memorized home locations, i.e., the geometric looping centers. At $h = 0.034$ to $h = 0.040$, this standard deviation more than doubles. For values below $h = 0.040$, no obvious trend can be recognized except for an optimum at $h = 0.034$.

The parameter k acts as a damping constant that regulates the overall activity of the CPU4 cells over time. Its increase or decrease is directly related to an increase or decrease in CPU4 activity over time (top left plot in figure 5.15). Moreover, if the damping constant is too high, it causes the integrators to forget the stored home location before reaching it (lower left plot in figure 5.15).

One more non-obvious effect becomes visible in the top right plot of figure 5.15. If k is set too low, the lateral CPU4 activities diverge. That is, one hemisphere

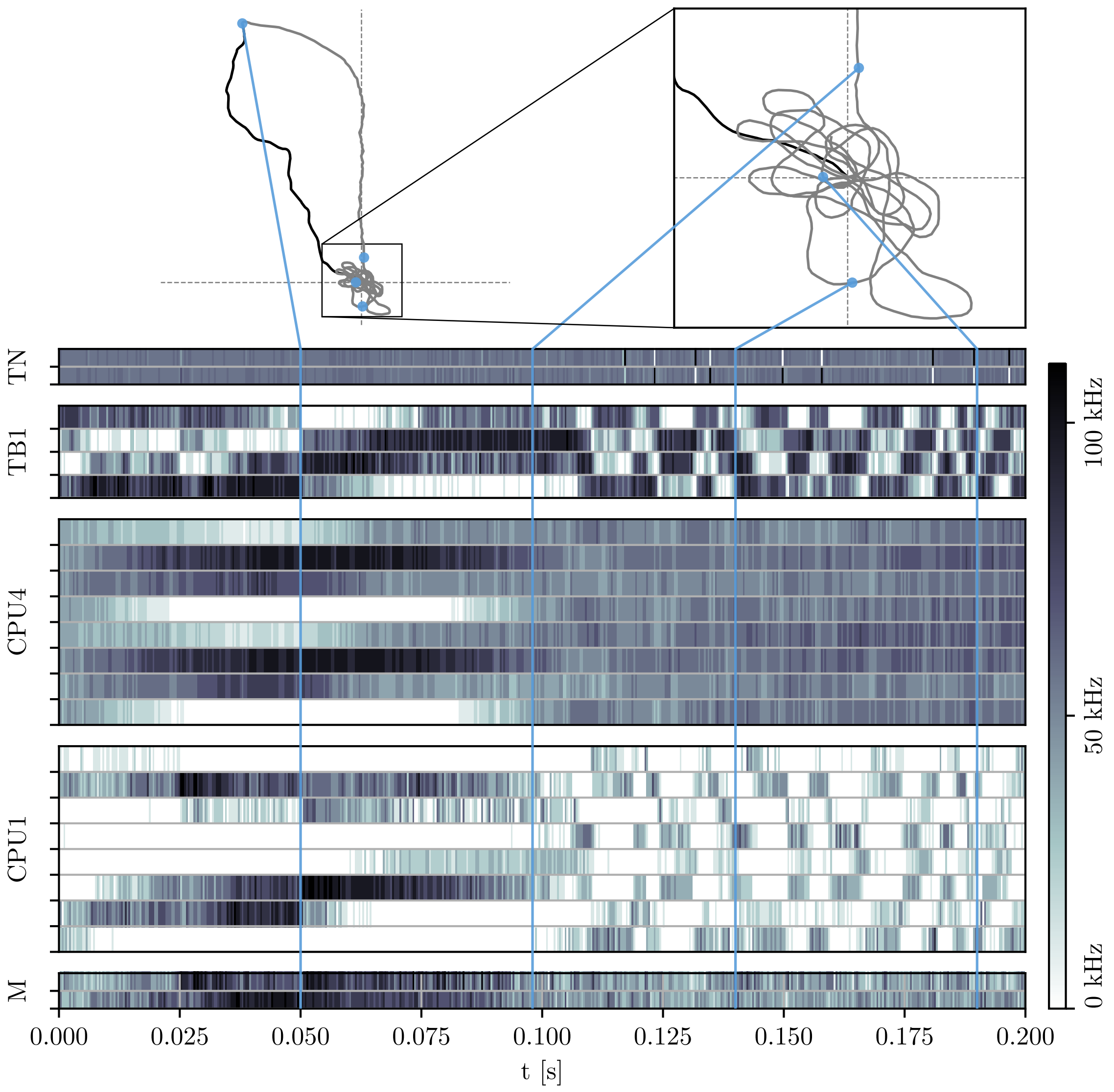


Figure 5.14: Network activity and trajectory. The spike rate is measured as the output spike count of a neuron within 100 µs intervals. This activity is color-coded over time. The maximum count is around 10, which corresponds to an instantaneous spike rate of 100 kHz. The top left plot shows the full trajectory consisting of spread-out (black) and return phase (gray). The top right plot zooms into the home area where the insects loops after returning. Four selected points in time are highlighted by the blue lines: the moment of return, shortly before reaching the home location, and two moments during the looping phase.

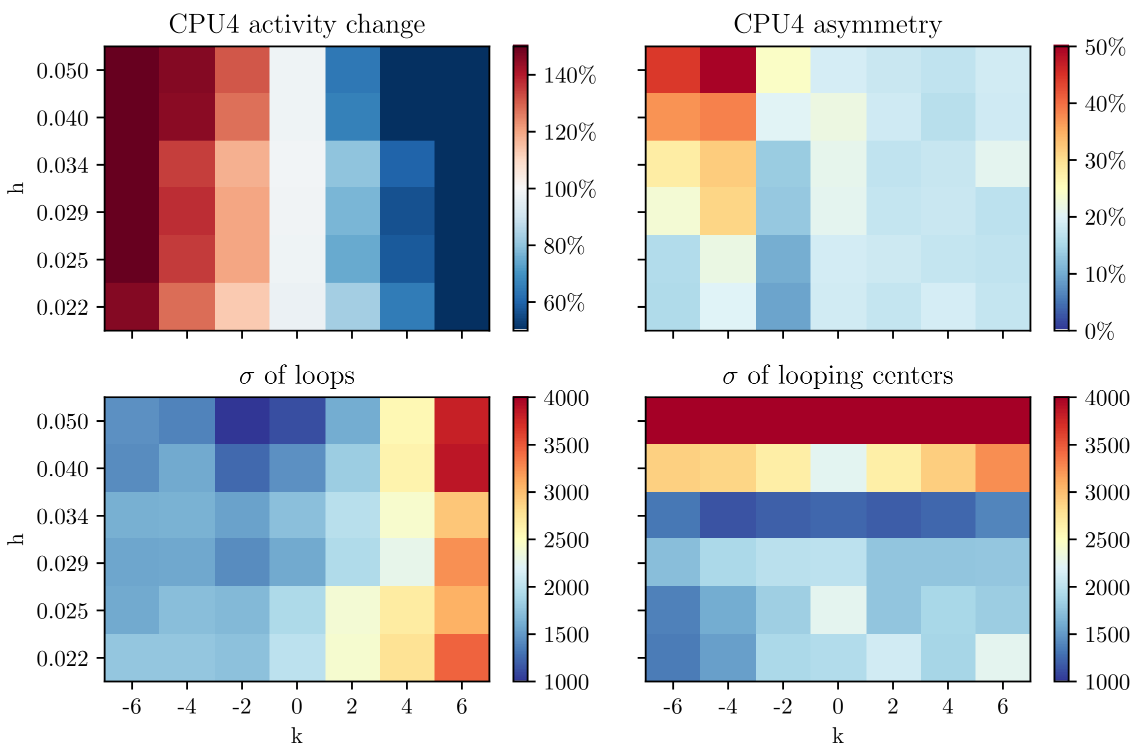


Figure 5.15: Effect of h and k (see equation (5.15)) on the network performance. Each data point reflects the statistic over 100 independent experimental runs for the respective parameter set. The top left plots shows the change in CPU4 activity relative to the spread-out phase ($t < t_{\text{return}}$) in percent. The plot CPU4 asymmetry depicts the difference in the left and right CPU4 activities relative the mean activity. σ of loops illustrates the mean standard deviation of the trajectories during $t > 2 \cdot t_{\text{return}}$, that is, during the looping phase. This measures how tight the individual looping radii are. Finally, σ of looping centers measures the standard deviation of the geometric centers of the individual trajectories during looping. This indicates, if the insect repeatedely returns to the same place over several iterations, or if it scatters.

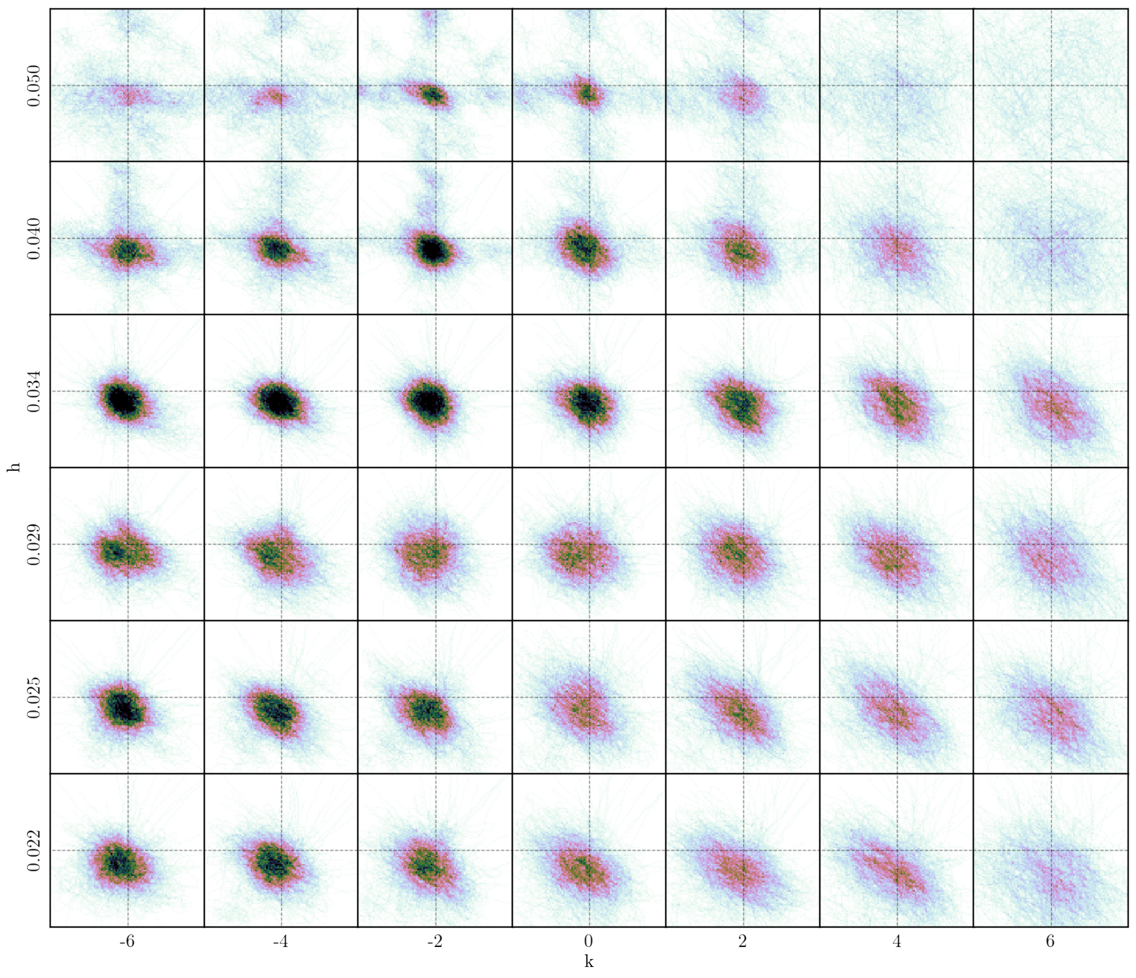


Figure 5.16: Effect of h and k (see equation (5.15)) on the network performance. Each square measures 12000×12000 and contains a histogram over the looping phase ($t > 2 \cdot t_{\text{return}}$) of 100 individual trajectories. The color encoding qualitatively represents the number of samples in each bin and goes from bright to dark.

becomes much more active than the other, causing the insect to rotate in tight circles rather than looping around in alternating left and right curves. The effect is self amplifying since the spontaneous rotation causes stronger TN input on the same CPU4 hemisphere that causes the looping. Thus, positive feedback is transferred from the integrators via the motor units and the odometer back to the integrators.

Figure 5.16 shows the same parameter sweep in a qualitative way to convey a little more intuition for the actual behavior of the path integrator. Moreover, appendix A.1 contains trajectory and activity plots of a few stereotypical cases.

The lowest standard deviation among the looping centers is in average achieved for $h = 0.034$. Given this constraint and considering all aspects of figure 5.15, the optimal values for k lie within -2 and 2. Since an increase in activity and asymmetry in the CPU4 cells is to be avoided, the parameter set $h = 0.034$ and $k = 2$ is selected for the next experiment.

5.7 Evolutionary optimization

To mitigate fixed-pattern noise effects and further optimize network performance, an evolution strategy with covariance matrix adaptation (Hansen and Ostermeier, 1996) is employed. The optimization parameters are the 26 synaptic weights $\boldsymbol{w}$ that project to the CPU1 population: TB1 → CPU1, CPU4 → CPU1 (inhibitory and excitatory), and CPU1 → M. To optimize for tight looping around the home position, the fitness f_i of an individual i is derived from its trajectory $\boldsymbol{x}_i(t)_{t=[2\cdot t_\mathrm{return}, t_\mathrm{stop})}$ during looping:

$$f_i = -\left\langle \left|\langle \boldsymbol{x}_i \rangle_t\right| + \left|\sqrt{\left\langle (\boldsymbol{x}_i - \langle \boldsymbol{x}_i \rangle)^2 \right\rangle_t}\right| \right\rangle_\mathrm{runs} \tag{5.20}$$

The first term is the time-averaged radial distance to the home location and the second term the time-averaged looping diameter. This sum is then additionally averaged over three runs with different outbound trajectories.

Instead of selecting a fixed number of best individuals of each generation, as in in the PlayPen experiment in section 4.8, the new mean parameter vector for the offspring population is derived from a weighted sum over the parameters of all individuals:

$$\boldsymbol{\mu} = \sum_i p_i \boldsymbol{w}_i \tag{5.21}$$

p_i is obtained as

$$p_i = \frac{\tilde{p}_i}{\sum_i \tilde{p}_i};\ \tilde{p}_i = f_i^{-8} \tag{5.22}$$

The power of eight overproportionally increases the fitness of successful individuals, while decreasing that of weak individuals. If the success of individuals is diverse, it exerts a strong evolutionary pressure. On the other hand, if the individuals are equally successful, the evolutionary pressure is released. Note that p_i is positive due

	μ_x/μ_y	σ_x/σ_y
primeval	-313/-774	1419/1398
optimized	-75/150	1125/1066
improvement	-76%/-81%	-21%/-24%

Table 5.1: Mean displacement and standard deviation before and after evolutionary optimization.

to the even exponential index. Moreover, small absolute values of f_i translate to a high p_i and vice versa.

The new population consists of 15 weight vectors drawn from a multivariate Gaussian distribution:

$$\boldsymbol{w}_i \sim \mathcal{N}(\boldsymbol{\mu}, \sigma \cdot \boldsymbol{\Sigma}) \tag{5.23}$$

Here, $\sigma = 0.3$ is a heuristically chosen step size and $\boldsymbol{\Sigma}$ is a covariance matrix derived from the previous generation

$$\boldsymbol{\Sigma} = \sum_i p_i \boldsymbol{d}_i \otimes \boldsymbol{d}_i \tag{5.24}$$

with $\boldsymbol{d}_i = \boldsymbol{w}_i - \boldsymbol{\mu}$. Like in section 4.8, the variance is increased into the direction of successful mutations. Typically, the optimization converges after approximately 200 generations.

To evaluate the robustness and performance of the network before and after the optimization, 1000 experiments were conducted for each case. The histogram over the return trajectories after $t = 2 \cdot t_{\text{return}}$ is shown in figure 5.17. The shape of this distribution reflects the average center and spread of the trajectories in the looping period. With the default weight settings, the network is already able to return home with sufficient accuracy, as the looping area overlaps the home location.

After evolutionary optimization, however, the center of looping is more aligned with the actual home location (0, 0) and the looping has a reduced spread and is therefore more precise. The results are summarized in table 5.1. The experiment shows that hardware fixed-pattern noise and inaccuracies can be compensated for by a weight-based optimization. Moreover, the entire evolution took 96 minutes. If it were to be carried out in the same manner in real-time without the acceleration factor, it would take more than two months instead.

5.8 Discussion and outlook

The presented experiments demonstrate a biologically constrained fully functional neuromorphic path integration agent. The agent's body and the environment are simulated on the digital on-chip coprocessor of BrainScaleS-2. The neural network is physically emulated on the same chip's analog neuromorphic core with digitally emulated sensoric inputs and motoric outputs attached to it via mixed-signal circuits.

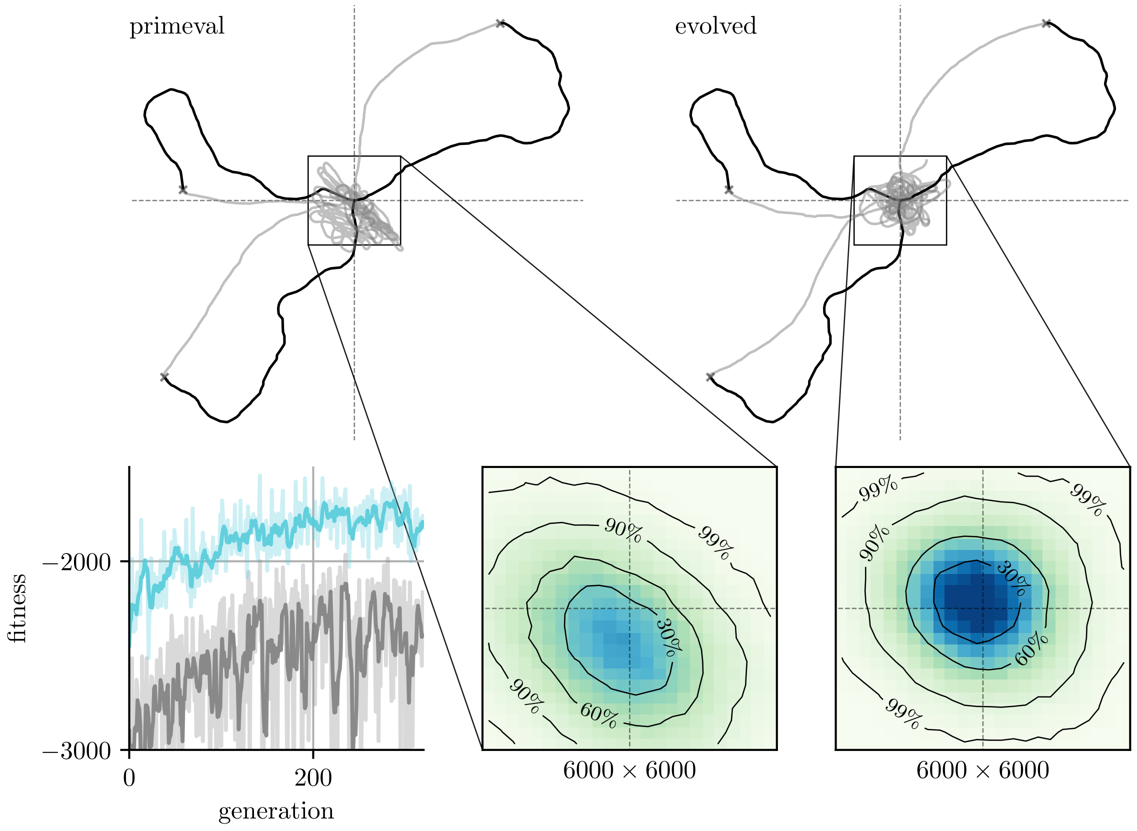


Figure 5.17: Statistical performance and evolutionary optimization. Three sample trajectories generated by the primeval (left) and the evolved network (right). Below are two 6000×6000 zooms into the center region that show the histograms over the data points of the looping phase of 1000 trajectories before and after optimization. The primeval network's center of looping is shifted to the lower left with a broad and elliptically deformed looping area. The evolved network on the other hand, is more centered and has a tighter and roundly shaped looping behavior. The lower left plot shows the population fitness (gray) and the fitness of the best three individuals (blue). For each, the faint line is the actual and the strong line the moving-average-filtered fitness for better visibility. The optimization converges after ~ 200 steps.

In contrast to previous experiments that only incorporated subcomponents of neural network models of insectoid or mammalian path integration into neuromorphic applications (Massoud and Horiuchi, 2010; Kreiser et al., 2018b,a, 2019b, 2020), the presented agent within its environment is realized on a stand-alone mixed-signal neuromorphic system.

The network is small enough to form a versatile building block for future insect-inspired systems that might integrate more and more neural subcomponents towards a neuromorphic implementation of entire insect nervous systems (Collins, 2019). The full-scale HICANN-X system, offers per-neuron random number generators to enable independent high-speed Poisson spike sources and a higher PPU clock speed. This relaxes the timing requirements on the PPU to a significant degree and, thus, leaves more space for further scheduling and control routines. For example, the CPU4 integrator populations can be instantiated repeatedly to allow for storing multiple locations at once. A hyper control routine (neural or digital) could then sequence the various target locations, mimicking bee's flower searching behavior.

Moreover, nothing prevents the setup from being connected to a physical robotic agent, like for example a flying multicopter. The reduced speed of such a robot can always be matched, since the neurons can operate in the rate limit and since the CPU4 update routines can be slowed down by an arbitrary factor.

6 Concluding remarks

"Until we see how many dimensions of behavior even a one-celled animal has, we won't be able to fully understand the behavior of more complicated animals."
– Richard Feynman, 'Surely You're Joking, Mr. Feynman!', 1997

The thesis started with some considerations about the relation between intelligence and interaction. It was concluded that the former without the latter typically implies purely symbolic intelligence, while the unity of both should naturally flow into an intelligence of action. The text continued with an argument for the exploration of small brains. This proposal was motivated primarily by the fact that the most significant findings in simulation neuroscience had recently been derived from simulations of small rather than large brains. Furthermore, by the paradoxical situation that even though small brains are still far from being well understood, research on the human brain is being conducted with much more momentum.

After a brief introduction to biology and neuromorphic systems, chapter 3 outlined the evolution and working principles of the hybrid BrainScaleS-2 system. The prototype as well as the full-scale chip versions and their test setups were introduced. One component, the CADC, was discussed in greater detail. Some accompanying experimental results then clarified the application areas and diversity of this parallel sensory component.

To address the desire for cybernetic interaction between an accelerated neuromorphic agent and mechanical reality, the next chapter 4 described the construction and functionality of an accelerated biomimetic robot. A prototype as well as an optimized final version were described and evaluated based on several experiments. Preliminary considerations already revealed that the realization of the concept would be at the limit of what is feasible. Experimentally this assumption was confirmed in that the robot is not yet able to reproduce the agility of small limbs but in any case that of typical agent trajectories. A demonstration of its speed advantages was provided in the form of an evolutionary optimization that consumed 2.5 hours in total. Had the process been executed in real-time, it would have taken three months instead. Since the platform is versatile, extensible, robust, and well documented, it is readily available for a wide range of future accelerated neuromorphic experiments.

The last chapter 5 dealt with functional aspects of insect brains with special reference to the central complex, a neural compound strongly related to spatial navigation. A functional model extracted from this brain area was modified to meet the requirements of BrainScaleS-2 and implemented successfully. The cybernetic insect reliably finds its way home with only 17 neurons. Again, it was shown how a genetic optimization of the network could be achieved within 1.5 hours instead of two months as it would have taken in biological real-time.

Very recently it has been remarked by Zhao et al. (2020) that "fully artificial agents" are yet missing in the context of neuromorphic hardware and that especially the development of "control and actuation [is] still lagging behind". Both aspects were addressed in this thesis and it may be justifiable that at least the lack of fully artificial agents was disproved. On the one hand, such an agent in terms of the neuromorphic bee has been introduced in chapter 5. On the other hand, chapter 4 contains the description of a fully neuromorphic robot that embodies a complete biomimetic cybernetic system comprising the control, actuation, feedback, and sensor components.

The second aspect, i.e. control and actuation, is certainly not mature yet but was also incorporated in both experiments. Moreover, everything satisfied the demands of an accelerated setup, which imposes far greater challenges than a system operating in biological real-time (see chapter 4). This difficulty has recently been made explicit by Yan et al. (2020), who questioned whether BrainScaleS-2 is at all capable of using "the neural activity of each time step [..] for the weight update[s]". Its capability to do so could be affirmed in chapter 5, as the accelerated insects rely on weight updates, which must be executed at a rate of no less than 10 kHz (10 Hz in biology) to reliably track their positions. Besides, the update of the actuators, which results from the spike count of the motor neurons, is also carried out at this rate. BrainScaleS-2 is therefore doubtlessly capable of processing and performing weight updates at timescales of merely a few inter-spike intervals.

Yan et al. (2020) also noted that it remains unclear so far how "a neural network running in accelerated time" can be interfaced with "robotic applications which require real time response". More clarity in this regard has been gained through the physically manifested accelerated robot described in chapter 4. This chapter also contains further descriptions of how the presented simple setup can potentially be extended to comprise more capable sets of both sensors and actuators. It was also noted there that unlike higher speeds, lower speeds are always accessible to spiking neuromorphic substrates when operating in the rate limit. Real-time responses can therefore straight-forwardly be achieved by accelerated hardware.

From a biomimetic point of view, continuously operating neuromorphic substrates integrate much more naturally into biologically derived cybernetic situations than discretely operating systems do. Although this is the case, the networks are much more often adapted to the working conditions of the latter, than the working conditions are adapted to the intrinsics of the neural networks. At present, this seems to be driven by the tremendous recent successes in machine learning which often involve batch-like evaluations of massive data sets. While until a few years ago spiking neural networks were predominantly simulated or emulated to increase the understanding of animal nervous systems, they are currently often forced to comply with the principles of machine learning. To achieve more competitive classification results, they are being exposed to monstrous series of artificial images. Most animals, including the author, would perceive this as torture of the most terrible kind.

The author, therefore, expresses his sincere hope that this trend will flow back into a more biological direction at some point in the near future and that neuromorphic organisms will be guided into more natural habitats. Small artificial brains with less than a thousand neurons could first be (re)naturalized to eventually set the process in motion. Maybe this thesis can contribute a little piece to this.

List of coauthored publications

This section lists the coauthored publications, clarifying which are incorporated in this document, and to what extent.

1. **Insectoid path integration on an accelerated neuromorphic substrate**
   Korbinian Schreiber, Timo Wunderlich, Philipp Spilger, Sebastian Billaudelle, Benjamin Cramer, Yannik Stradmann, Christian Pehle, Eric Müller, Mihai A. Petrovici, Johannes Schemmel, and Karlheinz Meier *(in preparation)*

   The author conceptualized, implemented, and evaluated all experimental aspects of this publication. Timo Wunderlich contributed to software simulations which preceded the hardware implementation. Philipp Spilger offered significant consulting and support concerning the PPU compiler. Parts of the calibration routines (sections 5.5.1 and 5.5.2) are based on previous work by Sebastian Billaudelle, Benjamin Cramer, and Yannik Stradmann. The content is part of chapter 5.

2. **Training spiking multi-layer networks with surrogate gradients on an analog neuromorphic substrate**
   Benjamin Cramer, Sebastian Billaudelle, Simeon Kanya, Aron Leibfried, Andreas Grübl, Vitali Karasenko, Christian Pehle, Korbinian Schreiber, Yannik Stradmann, Johannes Weis, Johannes Schemmel, Friedemann Zenke *arXiv preprint arXiv:2006.07239*

   The author contributed to this publication in terms of the xBoard test setup (section 3.4) and the CADC (section 3.5) which facilitates both, training and classification, in this learning experiment. Results from this publication are incorporated in section 3.5.

3. **Inference with Artificial Neural Networks on Analog Neuromorphic Hardware**
   Johannes Weis, Philipp Spilger, Sebastian Billaudelle, Yannik Stradmann, Arne Emmel, Eric Müller, Oliver Breitwieser, Andreas Grübl, Joscha Ilmberger, Vitali Karasenko, Mitja Kleider, Christian Mauch, Korbinian Schreiber, Johannes Schemmel *arXiv preprint arXiv:2006.13177*

   The author contributed to this publication in terms of the xBoard test setup (section 3.4) and the CADC (section 3.5) which is involved in both, training and classification. No content of this publication is presented in this thesis.

4. **Closed-loop experiments on the BrainScaleS-2 architecture**
   Korbinian Schreiber, Timo Wunderlich, Christian Pehle, Mihai A. Petrovici, Johannes Schemmel, Karlheinz Meier *Neuro-Inspired Computational Elements Workshop (NICE), 2020, Heidelberg, Germany*

This review-like publication includes aspects of chapters 4 and 5.

5. **Structural plasticity on an accelerated analog neuromorphic hardware system**
   Sebastian Billaudelle, Benjamin Cramer, Mihai A. Petrovici, Korbinian Schreiber, David Kappel, Johannes Schemmel, Karlheinz Meier *Neural Networks 133 (2021): 11-20*

   The author contributed to this publication in terms of the HICANN-DLSv3 test setup (section 3.2) and the CADC (section 3.5) which is involved in both, training and classification.

6. **Versatile emulation of spiking neural networks on an accelerated neuromorphic substrate**
   Sebastian Billaudelle, Yannik Stradmann, Korbinian Schreiber, Benjamin Cramer, Andreas Baumbach, Dominik Dold, Julian Göltz, Akos F. Kungl, Timo C. Wunderlich, Andreas Hartel, Eric Müller, Oliver Breitwieser, Christian Mauch, Mitja Kleider, Andreas Grübl, David Stöckel, Christian Pehle, Arthur Heimbrecht, Philipp Spilger, Gerd Kiene, Vitali Karasenko, Walter Senn, Mihai A. Petrovici, Johannes Schemmel, Karlheinz Meier *2020 IEEE International Symposium on Circuits and Systems (ISCAS), 2020, Sevilla, Spain*

   This review-like publication contains aspects of chapter 5.

7. **Demonstrating Advantages of Neuromorphic Computation: A Pilot Study**
   Timo Wunderlich, Akos F. Kungl, Eric Müller, Andreas Hartel, Yannik Stradmann, Syed Ahmed Aamir, Andreas Grübl, Arthur Heimbrecht, Korbinian Schreiber, David Stöckel, Christian Pehle, Sebastian Billaudelle, Gerd Kiene, Christian Mauch, Johannes Schemmel, Karlheinz Meier, Mihai A. Petrovici *Frontiers in Neuroscience 13 (2019): 260*

   The author contributed to this publication with the HICANN-DLSv3 test setup (section 3.2).

7 Bibliography

Aamir, S. A., Müller, P., Hartel, A., Schemmel, J., and Meier, K. A highly tunable 65-nm CMOS LIF neuron for a large scale neuromorphic system. In *IEEE European Solid-State Circuits Conference (ESSCIRC)*, pages 71–74, Sept. 2016.

Aamir, S. A., Müller, P., Kriener, L., Kiene, G., Schemmel, J., and Meier, K. From LIF to AdEx Neuron Models: Accelerated Analog 65 nm CMOS Implementation. In *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 1–4. IEEE, October 2017.

Aamir, S. A., Stradmann, Y., Müller, P., Pehle, C., Hartel, A., Grübl, A., Schemmel, J., and Meier, K. An accelerated LIF neuronal network array for a large-scale mixed-signal neuromorphic architecture. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 65(12):4299–4312, 2018.

Abbott, L. F. Lapicque's introduction of the integrate-and-fire model neuron (1907). *Brain research bulletin*, 50(5-6):303–304, 1999.

Abeles, M. Synfire chains. *Scholarpedia*, 4(7):1441, 2009.

Adden, A., Wibrand, S., Pfeiffer, K., Warrant, E., and Heinze, S. The brain of a nocturnal migratory insect, the Australian Bogong moth. *Journal of Comparative Neurology*, 528(11):1942–1963, 2020.

Ahrens, M. B., Orger, M. B., Robson, D. N., Li, J. M., and Keller, P. J. Whole-brain functional imaging at cellular resolution using light-sheet microscopy. *Nature methods*, 10(5):413–420, 2013.

Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. *Molecular biology of the cell, 5th edn. Garland Science.* Taylor & Francis, 2007.

ams. AS5600 – Standard Board. 2014. URL `https://ams.com/documents/20143/36005/AS5600_UG000379_1-00.pdf/46941386-c22a-f4ad-7fc2-ce9c3bb4af4e`, Rev. 1-00.

ams. AS5600 – 12-Bit Programmable Contactless Potentiometer. 2018. URL `https://ams.com/documents/20143/36005/AS5600_DS000365_5-00.pdf/649ee61c-8f9a-20df-9e10-43173a3eb323`, Rev. 1-06.

Analog Devices. AD5308/AD5318/AD5328 – 2.5 V to 5.5 V Octal Voltage Output 8-/10-/12-Bit DACs in 16-Lead TSSOP. 2016. URL `https://www.analog.com/media/en/technical-documentation/data-sheets/AD5308_5318_5328.pdf`, Rev. F – 4/2011.

Arduino. Arduino Nano – User Manual. 2008. URL `https://www.arduino.cc/en/uploads/Main/ArduinoNanoManual23.pdf`, Rev. 2.3.

Arends, J. and Zeigler, H. P. Organization of the cerebellum in the pigeon (Columba livia): I. Corticonuclear and corticovestibular connections. *Journal of comparative neurology*, 306(2):221–244, 1991.

Azevedo, F. A., Carvalho, L. R., Grinberg, L. T., Farfel, J. M., Ferretti, R. E., Leite, R. E., Filho, W. J., Lent, R., and Herculano-Houzel, S. Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology*, 513(5):532–541, 2009.

Banterle, F., Artusi, A., Debattista, K., and Chalmers, A. *Advanced high dynamic range imaging.* CRC press, 2017.

Bashir, S., Ali, S., Ahmed, S., and Kakkar, V. Analog-to-digital converters: A comparative study and performance analysis. In *2016 International Conference on Computing, Communication and Automation (ICCCA)*, pages 999–1001. IEEE, 2016.

Beaulieu, C. and Colonnier, M. Number of neurons in individual laminae of areas 3B, 4γ, and 6aα of the cat cerebral cortex: a comparison with major visual areas. *Journal of Comparative Neurology*, 279(2):228–234, 1989.

Bellec, G., Scherr, F., Hajek, E., Salaj, D., Legenstein, R., and Maass, W. Biologically inspired alternatives to backpropagation through time for learning in recurrent neural nets. *arXiv preprint arXiv:1901.09049*, 2019.

Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., and Maass, W. A solution to the learning dilemma for recurrent networks of spiking neurons. *bioRxiv*, page 738385, 2020.

Bengtson, S. Origins and early evolution of predation. *The Paleontological Society Papers*, 8:289–318, 2002.

Benjamin, B. V., Gao, P., McQuinn, E., Choudhary, S., Chandrasekaran, A. R., Bussat, J.-M., Alvarez-Icaza, R., Arthur, J. V., Merolla, P. A., and Boahen, K. Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proceedings of the IEEE*, 102(5):699–716, 2014.

Bernstein, J. Ueber den zeitlichen Verlauf der negativen Schwankung des Nervenstroms. *Archiv für die gesamte Physiologie des Menschen und der Tiere*, 1(1): 173–207, 1868.

Bernstein, J. Untersuchungen zur Thermodynamik der bioelektrischen Ströme. *Archiv für die gesamte Physiologie des Menschen und der Tiere*, 92(10-12):521–562, 1902.

Bi, G.-q. and Poo, M.-m. Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of neuroscience*, 18(24):10464–10472, 1998.

Billaudelle, S. Design and Implementation of a Short Term Plasticity Circuit for a 65 nm Neuromorphic Hardware System. Masterarbeit, Universität Heidelberg, 2017.

Bostrom, N. *Superintelligence*. Dunod, 2017.

Brette, R. Subjective physics. *arXiv preprint arXiv:1311.3129*, 2013.

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

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language Models are Few-Shot Learners. 2020.

Brüderle, D., Petrovici, M. A., Vogginger, B., Ehrlich, M., Pfeil, T., Millner, S., Grübl, A., Wendt, K., Müller, E., Schwartz, M.-O., et al. A comprehensive workflow for general-purpose neural modeling with highly configurable neuromorphic hardware systems. *Biological cybernetics*, 104(4-5):263–296, 2011.

Burrows, M. *The neurobiology of an insect brain*. Oxford University Press on Demand, 1996.

Byrne, J. H. and Kandel, E. R. Presynaptic facilitation revisited: state and time dependence. *Journal of Neuroscience*, 16(2):425–435, 1996.

Card, G. and Dickinson, M. H. Visually mediated motor planning in the escape response of drosophila. *Current Biology*, 18(17):1300–1307, 2018.

Cariani, P. On the Importance of Being Emergent. *Constructivist Foundations*, 5 (2), 2010.

Chapman, J. W., Reynolds, D. R., Smith, A. D., Riley, J. R., Pedgley, D. E., and Woiwod, I. P. High-altitude migration of the diamondback moth Plutella xylostella to the UK: a study using radar, aerial netting, and ground trapping. *Ecological Entomology*, 27(6):641–650, 2002.

Chen, Y., Zhao, H., Mao, J., Chirarattananon, P., Helbling, E. F., Hyun, N.-s. P., Clarke, D. R., and Wood, R. J. Controlled flight of a microrobot powered by soft artificial muscles. *Nature*, 575(7782):324–329, 2019.

Cheng, R., Mirza, K. B., and Nikolic, K. Neuromorphic robotic platform with visual input, processor and actuator, based on spiking neural networks. *Applied System Innovation*, 3(2):28, 2020.

Chiang, A.-S., Lin, C.-Y., Chuang, C.-C., Chang, H.-M., Hsieh, C.-H., Yeh, C.-W., Shih, C.-T., Wu, J.-J., Wang, G.-T., Chen, Y.-C., et al. Three-dimensional reconstruction of brain-wide wiring networks in Drosophila at single-cell resolution. *Current Biology*, 21(1):1–11, 2011.

Chiel, H. J. and Beer, R. D. The brain has a body: adaptive behavior emerges from interactions of nervous system, body and environment. *Trends in neurosciences*, 20(12):553–557, 1997.

Cimino, G. Reticular theory versus neuron theory in the work of Camillo Golgi., 1999.

Collett, T. S. and Collett, M. Path integration in insects. *Current opinion in neurobiology*, 10(6):757–762, 2000.

Collett, T. S. and Collett, M. Memory use in insect visual navigation. *Nature Reviews Neuroscience*, 3(7):542–552, 2002.

Collins, L. T. The case for emulating insect brains using anatomical "wiring diagrams" equipped with biophysical models of neuronal activity. *Biological Cybernetics*, 113(5-6):465–474, 2019.

Cox, B. R. and Krichmar, J. L. Neuromodulation as a robot controller. *IEEE Robotics & Automation Magazine*, 16(3):72–80, 2009.

Cramer, B., Billaudelle, S., Kanya, S., Leibfried, A., Grübl, A., Karasenko, V., Pehle, C., Schreiber, K., Stradmann, Y., Weis, J., et al. Training spiking multi-layer networks with surrogate gradients on an analog neuromorphic substrate. *arXiv preprint arXiv:2006.07239*, 2020a.

Cramer, B., Stöckel, D., Kreft, M., Wibral, M., Schemmel, J., Meier, K., and Priesemann, V. Control of criticality and computation in spiking neuromorphic networks with plasticity. *Nature Communications*, 11(1):1–11, 2020b.

Curcio, C. A., Sloan, K. R., Kalina, R. E., and Hendrickson, A. E. Human photoreceptor topography. *Journal of comparative neurology*, 292(4):497–523, 1990.

Dacke, M., Baird, E., Byrne, M., Scholtz, C. H., and Warrant, E. J. Dung beetles use the Milky Way for orientation. *Current Biology*, 23(4):298–300, 2013.

Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., Choday, S. H., Dimou, G., Joshi, P., Imam, N., Jain, S., et al. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018.

Dayan, P. and Abbott, L. F. Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems. 2001.

DeFelipe, J. The dendritic spine story: an intriguing process of discovery. *Frontiers in neuroanatomy*, 9:14, 2015.

DeFelipe, J., Alonso-Nanclares, L., and Arellano, J. I. Microstructure of the neocortex: comparative aspects. *Journal of neurocytology*, 31(3-5):299–316, 2002.

Douglas, R., Mahowald, M., and Mead, C. Neuromorphic analogue VLSI. *Annual review of neuroscience*, 18(1):255–281, 1995.

Doya, K. and Uchibe, E. The cyber rodent project: Exploration of adaptive mechanisms for self-preservation and self-reproduction. *Adaptive Behavior*, 13(2): 149–160, 2005.

Dukas, R. Evolutionary biology of animal cognition. *Annu. Rev. Ecol. Evol. Syst.*, 35:347–374, 2004.

Edgecombe, G. D., Giribet, G., Dunn, C. W., Hejnol, A., Kristensen, R. M., Neves, R. C., Rouse, G. W., Worsaae, K., and Sørensen, M. V. Higher-level metazoan relationships: recent progress and remaining questions. *Organisms Diversity & Evolution*, 11(2):151–172, 2011.

Ejaz, N., Krapp, H. G., and Tanaka, R. J. Closed-loop response properties of a visual interneuron involved in fly optomotor control. *Frontiers in neural circuits*, 7:50, 2013.

El Hady, A. *Closed loop neuroscience*. Academic Press, 2016.

el Jundi, B., Warrant, E. J., Pfeiffer, K., and Dacke, M. Neuroarchitecture of the dung beetle central complex. *Journal of Comparative Neurology*, 526(16): 2612–2630, 2018.

Elston, G. N., Benavides-Piccione, R., and DeFelipe, J. The pyramidal cell in cognition: a comparative study in human and monkey. *Journal of Neuroscience*, 21(17):RC163–RC163, 2001.

Engert, F. Fish in the matrix: motor learning in a virtual world. *Frontiers in neural circuits*, 6:125, 2013.

Erdem, H. A practical fuzzy logic controller for sumo robot competition. In *International Conference on Pattern Recognition and Machine Intelligence*, pages 217–225. Springer, 2007.

Evan-Amos. Laptop-hard-drive-exposed, 2013. URL `https://commons.wikimedia.org/wiki/File:Laptop-hard-drive-exposed.jpg`. File: https://creativecommons.org/licenses/by-sa/3.0/legalcode.

Finger, S. *Origins of neuroscience: a history of explorations into brain function.* Oxford University Press, USA, 2001.

FitzHugh, R. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical journal*, 1(6):445, 1961.

Floreano, D., Zufferey, J.-C., Srinivasan, M. V., and Ellington, C. *Flying insects and robots.* Springer, 2009.

Fourcaud, N. and Brunel, N. Dynamics of the firing probability of noisy integrate-and-fire neurons. *Neural computation*, 14(9):2057–2110, 2002.

Frank, G. Pulse code communication, March 17 1953. US Patent 2,632,058.

Frémaux, N. and Gerstner, W. Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules. *Frontiers in neural circuits*, 9:85, 2016.

Frémaux, N., Sprekeler, H., and Gerstner, W. Functional requirements for reward-modulated spike-timing-dependent plasticity. *Journal of Neuroscience*, 30(40): 13326–13337, 2010.

Frémaux, N., Sprekeler, H., and Gerstner, W. Reinforcement learning using a continuous time actor-critic framework with spiking neurons. *PLoS Comput Biol*, 9(4):e1003024, 2013.

Friedmann, S. *A new approach to learning in neuromorphic hardware.* PhD thesis, Universität Heidelberg, 2013.

Friedmann, S., Schemmel, J., Grübl, A., Hartel, A., Hock, M., and Meier, K. Demonstrating hybrid learning in a flexible neuromorphic hardware system. *IEEE transactions on biomedical circuits and systems*, 11(1):128–142, 2016.

Frisch, K. v. Die Polarisation des Himmelslichtes als orientierender Faktor bei den Tänzen der Bienen. *Experientia*, 5(4):142–148, 1949.

Fuglevand, A. J., Macefield, V. G., and Bigland-Ritchie, B. Force-frequency and fatigue properties of motor units in muscles that control digits of the human hand. *Journal of neurophysiology*, 81(4):1718–1729, 1999.

Furber, S. B., Galluppi, F., Temple, S., and Plana, L. A. The spinnaker project. *Proceedings of the IEEE*, 102(5):652–665, 2014.

Geirhos, R., Janssen, D. H., Schütt, H. H., Rauber, J., Bethge, M., and Wichmann, F. A. Comparing deep neural networks against humans: object recognition when the signal gets weaker. *arXiv preprint arXiv:1706.06969*, 2017.

Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. *Neuronal dynamics: From single neurons to networks and models of cognition.* Cambridge University Press, 2014.

Goldman, D. E. Potential, impedance, and rectification in membranes. *The Journal of general physiology*, 27(1):37–60, 1943.

Grah, G. and Ronacher, B. Three-dimensional orientation in desert ants: context-independent memorisation and recall of sloped path segments. *Journal of Comparative Physiology A*, 194(6):517–522, 2008.

Grant, G. How the 1906 Nobel Prize in Physiology or Medicine was shared between Golgi and Cajal. *Brain research reviews*, 55(2):490–498, 2007.

Green, J., Adachi, A., Shah, K. K., Hirokawa, J. D., Magani, P. S., and Maimon, G. A neural circuit architecture for angular integration in Drosophila. *Nature*, 546 (7656):101–106, 2017.

Green, J., Vijayan, V., Pires, P. M., Adachi, A., and Maimon, G. Walking Drosophila aim to maintain a neural heading estimate at an internal goal angle. *Biorxiv*, page 315796, 2018.

Greenewalt, C. H. Dimensional relationships for flying animals. *Smithsonian miscellaneous collections*, 1962.

Grob, R., Fleischmann, P. N., and Rössler, W. Learning to navigate–how desert ants calibrate their compass systems. *Neuroforum*, 25(2):109–120, 2019.

Grutzendler, J., Kasthuri, N., and Gan, W.-B. Long-term dendritic spine stability in the adult cortex. *Nature*, 420(6917):812–816, 2002.

gwern.net and GPT-3. GPT-3 Creative Fiction, 2020. URL `https://www.gwern.net/GPT-3`.

Güttler, G. M. *Achieving a Higher Integration Level of Neuromorphic Hardware using Wafer Embedding.* PhD thesis, Universität Heidelberg, November 2017.

Haken, H. Synergetics. *Naturwissenschaften*, 67(3):121–128, 1980.

Haken, H. Hermann Haken: Synergetik: Selbstorganisation in komplexen Systemen. `https://youtu.be/l3rnHiweKzM`, 2018. [Online; accessed 23-September-2020].

Haken, H. and Haken-Krell, M. *Entstehung von biologischer Information und Ordnung.* Wissenschaftliche Buchgesellschaft, 1989.

Haluszczynski, A., Aumeier, J., Herteux, J., and Räth, C. Reducing network size and improving prediction stability of reservoir computing. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(6):063136, 2020.

Hansen, N. and Ostermeier, A. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of IEEE international conference on evolutionary computation*, pages 312–317. IEEE, 1996.

Hartel, A. *Implementation and Characterization of Mixed-Signal Neuromorphic ASICs* . PhD thesis, Universität Heidelberg, 2016.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

Healy, K., McNally, L., Ruxton, G. D., Cooper, N., and Jackson, A. L. Metabolic rate and body size are linked with perception of temporal information. *Animal behaviour*, 86(4):685–696, 2013.

Hebb, D. O. *The organization of behavior: a neuropsychological theory*. J. Wiley; Chapman & Hall, 1949.

Heinze, S. and Homberg, U. Maplike representation of celestial E-vector orientations in the brain of an insect. *Science*, 315(5814):995–997, 2007.

Heinze, S. and Homberg, U. Linking the input to the output: new sets of neurons complement the polarization vision network in the locust central complex. *Journal of Neuroscience*, 29(15):4911–4921, 2009.

Heinze, S. and Reppert, S. M. Sun compass integration of skylight cues in migratory monarch butterflies. *Neuron*, 69(2):345–358, 2011.

Heinze, S. and Reppert, S. M. Anatomical basis of sun compass navigation I: the general layout of the monarch butterfly brain. *Journal of Comparative Neurology*, 520(8):1599–1628, 2012.

Herculano-Houzel, S., Mota, B., and Lent, R. Cellular scaling rules for rodent brains. *Proceedings of the National Academy of Sciences*, 103(32):12138–12143, 2006.

Hock, M. *Modern semiconductor technologies for neuromorphic hardware*. PhD thesis, Universität Heidelberg, July 2014.

Hodgkin, A. L. and Huxley, A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952a.

Hodgkin, A. L. and Huxley, A. F. The components of membrane conductance in the giant axon of Loligo. *The Journal of physiology*, 116(4):473, 1952b.

Hodgkin, A. L. and Huxley, A. F. Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo. *The Journal of physiology*, 116(4):449, 1952c.

Hodgkin, A. L. and Huxley, A. F. The dual effect of membrane potential on sodium conductance in the giant axon of Loligo. *The Journal of physiology*, 116(4):497, 1952d.

Homberg, U., Heinze, S., Pfeiffer, K., Kinoshita, M., and El Jundi, B. Central neural coding of sky polarization in insects. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 366(1565):680–687, 2011.

Huang, K.-H., Rupprecht, P., Frank, T., Kawakami, K., Bouwmeester, T., and Friedrich, R. W. A virtual reality system to analyze neural activity and behavior in adult zebrafish. *Nature Methods*, 17(3):343–351, 2020.

Hubel, D. H. *Eye, brain, and vision.* Scientific American Library/Scientific American Books, 1995.

Huber, R. and Knaden, M. Egocentric and geocentric navigation during extremely long foraging paths of desert ants. *Journal of Comparative Physiology A*, 201(6): 609–616, 2015.

Hurlock, E. C., McMahon, A., and Joho, R. H. Purkinje-cell-restricted restoration of Kv3. 3 function restores complex spikes and rescues motor coordination in Kcnc3 mutants. *Journal of Neuroscience*, 28(18):4640–4648, 2008.

Igarashi, J., Yamaura, H., and Yamazaki, T. Large-scale simulation of a layered cortical sheet of spiking network model using a tile partitioning method. *Frontiers in Neuroinformatics*, 13:71, 2019.

Ijspeert, A. J., Crespi, A., Ryczko, D., and Cabelguen, J.-M. From swimming to walking with a salamander robot driven by a spinal cord model. *science*, 315 (5817):1416–1420, 2007.

Indiveri, G., Linares-Barranco, B., Hamilton, T. J., Van Schaik, A., Etienne-Cummings, R., Delbruck, T., Liu, S.-C., Dudek, P., Häfliger, P., Renaud, S., et al. Neuromorphic silicon neuron circuits. *Frontiers in neuroscience*, 5:73, 2011.

Issa, J. Display power analysis and design guidelines to reduce power consumption. *Journal of Information Display*, 13(4):167–177, 2012.

Ito, K., Shinomiya, K., Ito, M., Armstrong, J. D., Boyan, G., Hartenstein, V., Harzsch, S., Heisenberg, M., Homberg, U., Jenett, A., et al. A systematic nomenclature for the insect brain. *Neuron*, 81(4):755–765, 2014.

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

Izhikevich, E. M. and FitzHugh, R. Fitzhugh-nagumo model. *Scholarpedia*, 1(9): 1349, 2006.

Jeffery, K. J. *The neurobiology of spatial behaviour.* Oxford University Press, USA, 2003.

Joly, J.-S., Recher, G., Brombin, A., Ngo, K., and Hartenstein, V. A conserved developmental mechanism builds complex visual systems in insects and vertebrates. *Current Biology*, 26(20):R1001–R1009, 2016.

Jonas, J. B., Schmidt, A. M., Müller-Bergh, J., Schlötzer-Schrehardt, U., and Naumann, G. Human optic nerve fiber count and optic disc size. *Investigative ophthalmology & visual science*, 33(6):2012–2018, 1992.

Jordan, J., Ippen, T., Helias, M., Kitayama, I., Sato, M., Igarashi, J., Diesmann, M., and Kunkel, S. Extremely scalable spiking neuronal network simulation code: from laptops to exascale computers. *Frontiers in neuroinformatics*, 12:2, 2018.

Jussios, L. A Precise Rotary Position Sensor for the PlayPen, 6 2018.

Kandel, E. R., Schwartz, J. H., Jessell, T. M., of Biochemistry, D., Jessell, M. B. T., Siegelbaum, S., and Hudspeth, A. *Principles of neural science*, volume 5. McGraw-hill New York, 2013.

Kandel, E. R. *Cellular basis of behavior: An introduction to behavioral neurobiology.* W. H. Freeman, 1976.

Karasenko, V. *Von Neumann bottlenecks in non-von Neumann computing architectures.* PhD thesis, Universität Heidelberg, May 2020.

Keller, D., Erö, C., and Markram, H. Cell densities in the mouse brain: a systematic review. *Frontiers in neuroanatomy*, 12:83, 2018.

Kempter, R., Gerstner, W., and Van Hemmen, J. L. Hebbian learning and spiking neurons. *Physical Review E*, 59(4):4498, 1999.

Kernell, D. *The motoneurone and its muscle fibres.* Oxford University Press, 2006.

Kernighan, B. W., Ritchie, D. M., et al. *The C programming language*, volume 2. Prentice Hall Professional Technical Reference, 1988.

Khurana, T. R. and Sane, S. P. Airflow and optic flow mediate antennal positioning in flying honeybees. *Elife*, 5:e14449, 2016.

Kiene, G. Mixed-Signal Neuron and Readout Circuits for a Neuromorphic System. Masterthesis, Universität Heidelberg, 2017.

Kleider, M. *Neuron Circuit Characterization in a Neuromorphic System.* PhD thesis, Universität Heidelberg, 2017.

Knight, J. C. and Nowotny, T. GPUs outperform current HPC and neuromorphic solutions in terms of speed and energy when simulating a highly-connected cortical model. *Frontiers in neuroscience*, 12:941, 2018.

Knight, J. C., Sakhapov, D., Domcsek, N., Dewar, A. D., Graham, P., Nowotny, T., and Philippides, A. Insect-Inspired Visual Navigation On-Board an Autonomous Robot: Real-World Routes Encoded in a Single Layer Network. In *Artificial Life Conference Proceedings*, pages 60–67. MIT Press, 2019.

Koch, C. *The Feeling of Life Itself: Why Consciousness is Widespread But Can't be Computed.* Mit Press, 2019.

Kreiser, R., Cartiglia, M., Martel, J. N., Conradt, J., and Sandamirskaya, Y. A neuromorphic approach to path integration: a head-direction spiking neural network with vision-driven reset. In *2018 IEEE international symposium on circuits and systems (ISCAS)*, pages 1–5. IEEE, 2018a.

Kreiser, R., Renner, A., Sandamirskaya, Y., and Pienroj, P. Pose estimation and map formation with spiking neural networks: towards neuromorphic slam. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2159–2166. IEEE, 2018b.

Kreiser, R., Sandamirskaya, Y., et al. Error-driven learning for self-calibration in a neuromorphic path integration system. In *Robust Artificial Intelligence for Neurorobotics (RAI-NR Workshop 2019)*, 2019a.

Kreiser, R., Waibel, G., Sandamirskaya, Y., et al. Self-calibration and learning on chip: towards neuromorphic robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2019)*, 2019b.

Kreiser, R., Leite, V. R., Serhan, B., Bartolozzi, C., Glover, A., Sandamirskaya, Y., et al. An On-chip Spiking Neural Network for Estimation of the Head Pose of the iCub Robot. *Frontiers in Neuroscience*, 2020.

Kunkel, S., Schmidt, M., Eppler, J. M., Plesser, H. E., Masumoto, G., Igarashi, J., Ishii, S., Fukai, T., Morrison, A., Diesmann, M., et al. Spiking network simulation code for petascale computers. *Frontiers in neuroinformatics*, 8:78, 2014.

LadyofHats. Complete neuron cell diagram, 2007. URL `https://commons.wikimedia.org/wiki/File:Complete_neuron_cell_diagram_en.svg`. https://creativecommons.org/publicdomain/zero/1.0/legalcode.

Lapicque, L. Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization. *Journal de Physiologie et de Pathologie Generalej*, 9: 620–635, 1907.

Le Moël, F., Stone, T., Lihoreau, M., Wystrach, A., and Webb, B. The central complex as a potential substrate for vector based navigation. *Frontiers in psychology*, 10:690, 2019.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lee, J. H., Delbruck, T., and Pfeiffer, M. Training deep spiking neural networks using backpropagation. *Frontiers in neuroscience*, 10:508, 2016.

Lem, S. *Summa technologiae*. Suhrkamp Verlag, 1981.

Leng, L., Martel, R., Breitwieser, O., Bytschok, I., Senn, W., Schemmel, J., Meier, K., and Petrovici, M. A. Spiking neurons with short-term synaptic plasticity form superior generative networks. *Scientific reports*, 8(1):1–11, 2018.

Liu, J. and Zhang, S. Multi-phase sumo maneuver learning. *Robotica*, 22(1):61, 2004.

Livi, P. and Indiveri, G. A current-mode conductance-based silicon neuron for address-event neuromorphic systems. In *2009 IEEE international symposium on circuits and systems*, pages 2898–2901. IEEE, 2009.

Lodish, H., Berk, A., Zipursky, S. L., Matsudaira, P., Baltimore, D., and Darnell, J. *Molecular Cell Biology.* W. H. Freeman, 4 edition, 2000. URL `https://www.ncbi.nlm.nih.gov/books/NBK21648/`.

Loukola, O. J., Perry, C. J., Coscos, L., and Chittka, L. Bumblebees show cognitive flexibility by improving on an observed complex behavior. *Science*, 355(6327): 833–836, 2017.

Ma, X., Hou, X., Edgecombe, G. D., and Strausfeld, N. J. Complex brain and optic lobes in an early Cambrian arthropod. *Nature*, 490(7419):258–261, 2012.

Manoonpong, P., Parlitz, U., and Wörgötter, F. Neural control and adaptive neural forward models for insect-like, energy-efficient, and adaptable locomotion of walking machines. *Frontiers in neural circuits*, 7:12, 2013.

Markram, H., Muller, E., Ramaswamy, S., Reimann, M. W., Abdellah, M., Sanchez, C. A., Ailamaki, A., Alonso-Nanclares, L., Antille, N., Arsever, S., et al. Reconstruction and simulation of neocortical microcircuitry. *Cell*, 163(2):456–492, 2015.

Masoli, S., Tognolina, M., Laforenza, U., Moccia, F., and D'Angelo, E. Parameter tuning differentiates granule cell subtypes enriching transmission properties at the cerebellum input stage. *Communications biology*, 3(1):1–12, 2020.

Massoud, T. M. and Horiuchi, T. K. A neuromorphic VLSI head direction cell system. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 58(1): 150–163, 2010.

maxim integrated. MAX11612 – MAX11617 – Low-Power, 4-/8-/12-Channel, I2C, 12-Bit ADCs in Ultra-Small Packages. 2012. URL `https://datasheets.maximintegrated.com/en/ds/MAX11612-MAX11617.pdf`, Rev. A – 5/2012.

maxim integrated. MAX4206 – Precision Transimpedance Logarithmic Amplifier with Over 5 Decades of Dynamic Range. 2015. URL `https://datasheets.maximintegrated.com/en/ds/MAX4206.pdf`, Rev. 2.

McClintock, H., Temel, F. Z., Doshi, N., Koh, J.-s., and Wood, R. J. The milliDelta: A high-bandwidth, high-precision, millimeter-scale Delta robot. *Science Robotics*, 3(14):eaar3018, 2018.

McGregor, R. ロボット相撲. `https://youtu.be/QCqxOzKNFks`, 2019. [Online; accessed 03-August-2020].

Mead, C. *Analog VLSI and neural systems.* Addison-Wesley Longman Publishing Co., Inc., 1989.

Mead, C. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10): 1629–1636, 1990.

Menzel, R. and Giurfa, M. Cognitive architecture of a mini-brain: the honeybee. *Trends in cognitive sciences*, 5(2):62–71, 2001.

Merolla, P. A., Arthur, J. V., Alvarez-Icaza, R., Cassidy, A. S., Sawada, J., Akopyan, F., Jackson, B. L., Imam, N., Guo, C., Nakamura, Y., et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

Michel, J.-B., Shen, Y. K., Aiden, A. P., Veres, A., Gray, M. K., Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., Orwant, J., et al. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182, 2011.

Microchip Technology Inc. MCP4922 – 12-Bit Dual Voltage Output Digital-to-Analog Converter with SPI Interface. 2010. URL `http://ww1.microchip.com/downloads/en/DeviceDoc/22250A.pdf`, Rev. A – 4/2010.

Microchip Technology Inc. 24AA02UID – 2K I2C Serial EEPROMs with Unique 32-bit Serial Number. 2013. URL `https://ww1.microchip.com/downloads/en/DeviceDoc/20005202A.pdf`, Rev. A – 5/2013.

Milde, M. B., Blum, H., Dietmüller, A., Sumislawska, D., Conradt, J., Indiveri, G., and Sandamirskaya, Y. Obstacle avoidance and target acquisition for robot navigation using a mixed signal analog/digital neuromorphic processing system. *Frontiers in neurorobotics*, 11:28, 2017.

Möller, R., Lambrinos, D., Roggendorf, T., Pfeifer, R., and Wehner, R. Insect strategies of visual homing in mobile robots. In *Proceedings of the Computer Vision and Mobile Robotics Workshop CVMR*, volume 98, pages 75–82. Citeseer, 2001.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

Moradi, S., Qiao, N., Stefanini, F., and Indiveri, G. A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (DYNAPs). *IEEE transactions on biomedical circuits and systems*, 12 (1):106–122, 2017.

Mountcastle, V. B., Talbot, W. H., Kornhuber, H. H., et al. The neural transformation of mechanical stimuli delivered to the monkey's hand. *Touch, heat and pain*, pages 325–345, 1966.

Müller, E. C. *Novel operation modes of accelerated neuromorphic hardware.* PhD thesis, Ruprecht-Karls-Universität Heidelberg, 2015.

Müller, M. G., Papadimitriou, C. H., Maass, W., and Legenstein, R. A model for structured information representation in neural networks of the brain. *Eneuro*, 7 (3), 2020.

Müller, M., Homberg, U., and Kühn, A. Neuroarchitecture of the lower division of the central body in the brain of the locust (Schistocerca gregaria). *Cell and tissue research*, 288(1):159–176, 1997.

Nagumo, J., Arimoto, S., and Yoshizawa, S. An active pulse transmission line simulating nerve axon. *Proceedings of the IRE*, 50(10):2061–2070, 1962.

Napper, R. and Harvey, R. Number of parallel fiber synapses on an individual Purkinje cell in the cerebellum of the rat. *Journal of Comparative Neurology*, 274 (2):168–177, 1988.

Neckar, A., Fok, S., Benjamin, B. V., Stewart, T. C., Oza, N. N., Voelker, A. R., Eliasmith, C., Manohar, R., and Boahen, K. Braindrop: A mixed-signal neuromorphic architecture with a dynamical systems-based programming model. *Proceedings of the IEEE*, 107(1):144–164, 2018.

Neftci, E., Das, S., Pedroni, B., Kreutz-Delgado, K., and Cauwenberghs, G. Event-driven contrastive divergence for spiking neuromorphic systems. *Frontiers in neuroscience*, 7:272, 2014.

Neuser, K., Triphan, T., Mronz, M., Poeck, B., and Strauss, R. Analysis of a spatial orientation memory in Drosophila. *Nature*, 453(7199):1244, 2008.

Nexperia. 74HC14; 74HCT14. 2020. URL `https://assets.nexperia.com/documents/data-sheet/74HC_HCT14.pdf`, Rev. 8.

Nguyen, A., Yosinski, J., and Clune, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

Nityananda, V., Skorupski, P., and Chittka, L. Can bees see at a glance? *Journal of Experimental Biology*, 217(11):1933–1939, 2014.

Nyquist, H. Certain topics in telegraph transmission theory. *American Institute of Electrical Engineers, Transactions of the*, 47(2):617–644, 1928.

Oroquieta, J. D. *El jardín de la neurología: sobre lo bello, el arte y el cerebro.* Boletín Oficial del Estado, 2014.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

Parker, A. In the blink of an eye: how vision sparked the big bang of evolution. 2003.

Petrovici, M. A. *Form versus function: theory and models for neuronal substrates*. Springer, 2016.

Piccolino, M. Luigi Galvani and animal electricity: two centuries after the foundation of electrophysiology. *Trends in neurosciences*, 20(10):443–448, 1997.

Potjans, T. C. and Diesmann, M. The cell-type specific cortical microcircuit: relating structure and activity in a full-scale spiking network model. *Cerebral cortex*, 24 (3):785–806, 2014.

Potter, S. M., El Hady, A., and Fetz, E. E. Closed-loop neuroscience and neuroengineering. *Frontiers in neural circuits*, 8:115, 2014.

Prinzler, M. Accelerated neurorobotics on the HICANN-DLS system, 10 2018.

Qiao, N., Mostafa, H., Corradi, F., Osswald, M., Stefanini, F., Sumislawska, D., and Indiveri, G. A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128K synapses. *Frontiers in neuroscience*, 9:141, 2015.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language Models are Unsupervised Multitask Learners. 2019.

Reimann, M. W., Anastassiou, C. A., Perin, R., Hill, S. L., Markram, H., and Koch, C. A biophysically detailed model of neocortical local field potentials predicts the critical role of active membrane currents. *Neuron*, 79(2):375–390, 2013.

Ronacher, B. Path integration as the basic navigation mechanism of the desert ant Cataglyphis fortis (Forel, 1902)(Hymenoptera: Formicidae). *Myrmecological News*, 11:53–62, 2008.

Rucci, M., Bullock, D., and Santini, F. Integrating robotics and neuroscience: brains for robots, bodies for brains. *Advanced Robotics*, 21(10):1115–1129, 2007.

Sacramento, J., Costa, R. P., Bengio, Y., and Senn, W. Dendritic cortical microcircuits approximate the backpropagation algorithm. In *Advances in neural information processing systems*, pages 8721–8732, 2018.

Sandamirskaya, Y., Richter, M., and Schöner, G. A neural-dynamic architecture for behavioral organization of an embodied agent. In *2011 IEEE International Conference on Development and Learning (ICDL)*, volume 2, pages 1–7. IEEE, 2011.

Sarma, G. P., Lee, C. W., Portegys, T., Ghayoomie, V., Jacobs, T., Alicea, B., Cantarelli, M., Currie, M., Gerkin, R. C., Gingell, S., et al. OpenWorm: overview and recent advances in integrative biological simulation of Caenorhabditis elegans. *Philosophical Transactions of the Royal Society B*, 373(1758):20170382, 2018.

Schemmel, J., Fieres, J., and Meier, K. Wafer-scale integration of analog neural networks. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 431–438. IEEE, 2008.

Schemmel, J., Briiderle, D., Griibl, A., Hock, M., Meier, K., and Millner, S. A wafer-scale neuromorphic hardware system for large-scale neural modeling. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 1947–1950. IEEE, 2010.

Schemmel, J., Kriener, L., Müller, P., and Meier, K. An accelerated analog neuromorphic hardware system emulating NMDA-and calcium-based non-linear dendrites. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2217–2226. IEEE, 2017.

Schemmel, J., Billaudelle, S., Dauer, P., and Weis, J. Accelerated Analog Neuromorphic Computing. *arXiv preprint arXiv:2003.11996*, 2020.

Scheuerlein, H., Henschke, F., and Köckerling, F. Wilhelm von Waldeyer-Hartz—A Great Forefather: His Contributions to Anatomy with Particular Attention to "His" Fascia. *Frontiers in surgery*, 4:74, 2017.

Schmidhuber, J. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

Schneider-Mizell, C. M., Gerhard, S., Longair, M., Kazimiers, T., Li, F., Zwart, M. F., Champion, A., Midgley, F. M., Fetter, R. D., Saalfeld, S., et al. Quantitative neuroanatomy for connectomics in Drosophila. *Elife*, 5:e12059, 2016.

Schreiber, K. CADC – An 8 bit, scalable, ultra-low-power ADC for synaptic correlation measurements. In *Circuits and methods for VLSI design: Weekly seminar of the Heidelberg ASIC-Laboratory*. Heidelberg University, July 2016.

Seelig, J. D. and Jayaraman, V. Neural dynamics for landmark orientation and angular path integration. *Nature*, 521(7551):186–191, 2015.

Seyfarth, E.-A. Julius Bernstein (1839–1917): pioneer neurobiologist and biophysicist. *Biological cybernetics*, 94(1):2–8, 2006.

Shen, J., Ma, D., Gu, Z., Zhang, M., Zhu, X., Xu, X., Xu, Q., Shen, Y., and Pan, G. Darwin: a neuromorphic hardware co-processor based on spiking neural networks. *Science China Information Sciences*, 59(2):1–5, 2016.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of Go with deep neural networks and tree search. *nature*, 529 (7587):484–489, 2016.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Sjöström, J. and Gerstner, W. Spike-timing dependent plasticity. *Spike-timing dependent plasticity*, 35(0):0–0, 2010.

Smarandache-Wellmann, C. R. Arthropod neurons and nervous system. *Current Biology*, 26(20):R960–R965, 2016.

Sotavalta, O. Recordings of high wing-stroke and thoracic vibration frequency in some midges. *The Biological Bulletin*, 104(3):439–444, 1953.

SP9 partners, UHEI, UMAN, CNRS-UNIC, TUD, and KTH. *Neuromorphic Platform Specification*. Human Brain Project, 05 2020.

Srinivasan, M., Zhang, S., Lehrer, M., and Collett, T. Honeybee navigation en route to the goal: visual flight control and odometry. *Journal of Experimental Biology*, 199(1):237–244, 1996.

Stevens, C. F. and Zador, A. M. When is an integrate-and-fire neuron like a poisson neuron? In *Advances in neural information processing systems*, pages 103–109, 1996.

STMicroelectronics. STM32F103C8. 2015. URL `https://www.st.com/resource/en/datasheet/stm32f103c8.pdf`, Rev. 17 – 8/2015.

STMicroelectronics. L6234 – Three-phase motor driver. 2017. URL `https://www.st.com/resource/en/datasheet/l6234.pdf`, Rev. 11.

Stone, T., Webb, B., Adden, A., Weddig, N. B., Honkanen, A., Templin, R., Wcislo, W., Scimeca, L., Warrant, E., and Heinze, S. An anatomically constrained model for path integration in the bee brain. *Current Biology*, 27(20):3069–3085, 2017.

Strata, P. and Harvey, R. Dale's principle. *Brain research bulletin*, 50(5-6):349–350, 1999.

Su, J., Vargas, D. V., and Sakurai, K. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

Sundstrom, T., Murmann, B., and Svensson, C. Power dissipation bounds for high-speed Nyquist analog-to-digital converters. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 56(3):509–518, 2008.

Szakal, D., Honkanen, A., and Heinze, S. mg-Inter-LAL-DN1 Megalopta genalis, 2015. URL `https://hdl.handle.net/20.500.12158/NIN-0000095.1`. License: IBdb CC-BY, Source: insectbraindb.org.

Takemura, S., Bharioke, A., Lu, Z., Nern, A., Vitaladevuni, S., Rivlin, P. K., Katz, W. T., Olbris, D. J., Plaza, S. M., Winston, P., et al. A visual motion detection circuit suggested by Drosophila connectomics. *Nature*, 500(7461):175, 2013.

Takemura, S., Aso, Y., Hige, T., Wong, A., Lu, Z., Xu, C. S., Rivlin, P. K., Hess, H., Zhao, T., Parag, T., et al. A connectome of a learning and memory center in the adult Drosophila brain. *Elife*, 6:e26975, 2017.

Talbot, W. H., Darian-Smith, I., Kornhuber, H. H., and Mountcastle, V. B. The sense of flutter-vibration: comparison of the human capacity with response patterns of mechanoreceptive afferents from the monkey hand. *Journal of neurophysiology*, 31(2):301–334, 1968.

Texas Instruments. LMH6611/LMH6612 – Single Supply 345 MHz Rail-to-Rail Output Amplifiers. 2013. URL `https://www.ti.com/lit/gpn/LMH6611`, Rev. J.

Texas Instruments. DRV8833 Dual H-Bridge Motor Driver. 2015a. URL `https://www.ti.com/lit/gpn/DRV8833`, Rev. E – 3/2015.

Texas Instruments. INA219 – Zero-Drift, Bidirectional Current/Power Monitor With I^2C Interface. 2015b. URL `https://www.ti.com/lit/gpn/ina219`, Rev. G.

Texas Instruments. PGA11x Zerø-Drift Programmable Gain Amplifier With Mux. 2015c. URL `https://www.ti.com/lit/gpn/pga112`, Rev. C – 11/2015.

Texas Instruments. LMH6518 – 900 MHz, Digitally Controlled, Variable Gain Amplifier. 2016. URL `https://www.ti.com/lit/gpn/LMH6518`, Rev. D.

Texas Instruments. SN74LVC2T45 – Dual-Bit Dual-Supply Bus Transceiver With Configurable Voltage Translation. 2017. URL `https://www.ti.com/lit/gpn/sn74lvc2t45`, Rev. K – 10/2014.

Texas Instruments. TS5A3357 – Single 5-Ohm SP3T Analog Switch 5-V/3.3-V 3:1 Multiplexer/Demultiplexer. 2018a. URL `https://www.ti.com/lit/pdf/scds177`, Rev. B.

Texas Instruments. TXB0104 4-Bit Bidirectional Voltage-Level Translator with Auto-Direction Sensing and ś15-kV ESD Protection. 2018b. URL `https://www.ti.com/lit/gpn/TXB0104`, Rev. H.

Texas Instruments. TXB0108 8-Bit Bidirectional Voltage-Level Translator with Auto-Direction Sensing and 15-kV ESD Protection. 2018c. URL `https://www.ti.com/lit/gpn/TXB0108`, Rev. G.

Texas Instruments. ADC322x – Dual-Channel, 12-Bit, 25-MSPS to 125-MSPS, Analog-to-Digital Converters. 2019. URL `https://www.ti.com/lit/pdf/sbas672`, Rev. D – 7/2019.

Tootell, R. B., Switkes, E., Silverman, M. S., and Hamilton, S. L. Functional anatomy of macaque striate cortex. II. Retinotopic organization. *Journal of neuroscience*, 8(5):1531–1568, 1988.

Tootle, J. S. and Friedlander, M. J. Postnatal development of the spatial contrast sensitivity of X-and Y-cells in the kitten retinogeniculate pathway. *Journal of Neuroscience*, 9(4):1325–1340, 1989.

TOP500.org. Green500, 2020. URL `https://www.top500.org/lists/green500/2020/06/`.

Tsodyks, M., Pawelzik, K., and Markram, H. Neural networks with dynamic synapses. *Neural computation*, 10(4):821–835, 1998.

Tsodyks, M. V. and Markram, H. The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *Proceedings of the national academy of sciences*, 94(2):719–723, 1997.

Turing, A. M. Computing machinery and intelligence. *Mind*, 59(236):433, 1950.

Turner-Evans, D. B. and Jayaraman, V. The insect central complex. *Current Biology*, 26(11):R453–R457, 2016.

Unios Australia Pty Ltd. Driving the Flicker-Free Effect, 2019. URL `https://unios.com/wp-content/uploads/2016/06/UN_Driving-the-Flicker-Free-Effect_White-Paper-190205.pdf`. whitepaper.

van Albada, S. J., Rowley, A. G., Senk, J., Hopkins, M., Schmidt, M., Stokes, A. B., Lester, D. R., Diesmann, M., and Furber, S. B. Performance comparison of the digital neuromorphic hardware SpiNNaker and the neural network simulation software NEST for a full-scale cortical microcircuit model. *Frontiers in neuroscience*, 12:291, 2018.

Vannucci, L., Ambrosano, A., Cauli, N., Albanese, U., Falotico, E., Ulbrich, S., Pfotzer, L., Hinkel, G., Denninger, O., Peppicelli, D., et al. A visual tracking model implemented on the iCub robot as a use case for a novel neurorobotic toolkit integrating brain and physics simulation. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 1179–1184. IEEE, 2015.

Varga, A. G. and Ritzmann, R. E. Cellular basis of head direction and contextual cues in the insect brain. *Current Biology*, 26(14):1816–1828, 2016.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

VISHAY. BPW34 – Silicon PIN Photodiode. 2011. URL `https://www.vishay.com/docs/81521/bpw34.pdf`, Rev. 2.1.

Von Frisch, K. The dance language and orientation of bees. 1967.

Von Neumann, J. First Draft of a Report on the EDVAC. *IEEE Annals of the History of Computing*, 15(4):27–75, 1993.

Wagner, H. Flight performance and visual control of flight of the free-flying housefly (Musca domestica L.) I. Organization of the flight motor. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, 312(1158):527–551, 1986.

Waldeyer, W. Ueber einige neuere Forschungen im Gebiete der Anatomie des Centralnervensystems1. *DMW-Deutsche Medizinische Wochenschrift*, 17(44): 1213–1218, 1891.

Walsh, D. and Sved, A. Motor units, 2019. URL `https://commons.wikimedia.org/wiki/File:Motor_unit.png`. File: https://creativecommons.org/licenses/by-sa/4.0/legalcode.

Wang, Y.-F. and Hatton, G. I. Milk ejection burst-like electrical activity evoked in Supraoptic oxytocin neurons in slices from lactating rats. *Journal of neurophysiology*, 91(5):2312–2321, 2004.

Warrant, E. and Dacke, M. Visual navigation in nocturnal insects. *Physiology*, 31 (3):182–192, 2016.

Wehner, R. Desert ant navigation: how miniature brains solve complex tasks. *Journal of Comparative Physiology A*, 189(8):579–588, 2003.

Wehner, R. and Gehring, W. J. *Zoologie*. Georg Thieme Verlag, 2013.

Weis, J. Characterization and Calibration of Synaptic Plasticity on Neuromorphic Hardware. Bachelor, Universität Heidelberg, 2018.

Weis, J. Inference with Artificial Neural Networks on Neuromorphic Hardware. Master's thesis, Universität Heidelberg, 2020.

Weste, N. H. and Harris, D. *CMOS VLSI design: a circuits and systems perspective*. Pearson Education India, 2015.

Western Digital. WD VelociRaptor VR200M. 2010. URL `https://www.1000ordi.ch/wd-velociraptor-vr200m-sata-6-gb_s-450-gb-wd4500hlhx-54449_en.pdf`.

Wiener, N. *Cybernetics or Control and Communication in the Animal and the Machine*. MIT press, 1948.

Wikipedia. Robot-sumo — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Robot-sumo&oldid=909448568`, 2019. [Online; accessed 03-August-2020].

Wikipedia. PowerPC — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=PowerPC&oldid=975955517`, 2020a. [Online; accessed 06-September-2020].

Wikipedia. Three Gorges Dam — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Three%20Gorges%20Dam&oldid=976025481`, 2020b. [Online; accessed 04-September-2020].

Williams, R. W. and Herrup, K. The control of neuron number. *Annual review of neuroscience*, 11(1):423–453, 1988.

Wittlinger, M., Wehner, R., and Wolf, H. The ant odometer: stepping on stilts and stumps. *science*, 312(5782):1965–1967, 2006.

Wittmann, T. and Schwegler, H. Path integration—a network model. *Biological Cybernetics*, 73(6):569–575, 1995.

Wohlgemuth, S., Ronacher, B., and Wehner, R. Ant odometry in the third dimension. *Nature*, 411(6839):795–798, 2001.

Wunderlich, T., Kungl, A. F., Müller, E., Hartel, A., Stradmann, Y., Aamir, S. A., Grübl, A., Heimbrecht, A., Schreiber, K., Stöckel, D., et al. Demonstrating advantages of neuromorphic computation: a pilot study. *Frontiers in neuroscience*, 13:260, 2019.

Xu, C. S., Januszewski, M., Lu, Z., Takemura, S.-y., Hayworth, K., Huang, G., Shinomiya, K., Maitin-Shepard, J., Ackerman, D., Berg, S., et al. A connectome of the adult drosophila central brain. *BioRxiv*, 2020.

y Cajal, S. R. *Estructura de los centros nerviosos de las aves.* 1888.

Yakubowski, J. M., McMillan, G. A., and Gray, J. R. Background visual motion affects responses of an insect motion-sensitive neuron to objects deviating from a collision course. *Physiological Reports*, 4(10):e12801, 2016.

Yan, Y., Stewart, T. C., Choo, X., Vogginger, B., Partzsch, J., Hoeppner, S., Kelber, F., Eliasmith, C., Furber, S., and Mayr, C. Low-Power Low-Latency Keyword Spotting and Adaptive Control with a SpiNNaker 2 Prototype and Comparison with Loihi. *arXiv preprint arXiv:2009.08921v1*, 2020.

Zenke, F. and Ganguli, S. Superspike: Supervised learning in multilayer spiking neural networks. *Neural computation*, 30(6):1514–1541, 2018.

Zhao, C., Widmer, Y., Diegelmann, S., Petrovici, M., Sprecher, S., and Senn, W. Predictive olfactory learning in Drosophila. *bioRxiv*, 2019.

Zhao, J., Risi, N., Monforte, M., Bartolozzi, C., Indiveri, G., and Donati, E. Closed-loop spiking control on a neuromorphic processor implemented on the iCub. *arXiv preprint arXiv:2009.09081v1*, 2020.

Zwicker, E. and Fastl, H. *Psychoacoustics: Facts and models*, volume 22. Springer Science & Business Media, 2013.

Glossary

ADC analog-to-digital converter 3, 37, 38, 42, 43, 44, 47, 52, 54, 55, 58, 60, 62, 65, 66, 94, 103, 104, 106, 108, 111, 182

AdEx adaptive exponential integrate-and-fire 22, 41

ANNCORE analog neural network core 37, 38, 42, 43, 49, 50, 51, 182

ASIC application-specific integrated circuit 46, 67

ATP adenosine triphosphate 14

BrainScaleS accelerated analog large-scale neuromorphic system developed by Heidelberg University 2, 35, 36, 37, 49

BrainScaleS-1 1st generation of the BrainScaleS system 52

BrainScaleS-2 2nd generation of the BrainScaleS system 2, 3, 4, 8, 36, 37, 38, 49, 52, 67, 68, 69, 70, 73, 77, 78, 79, 80, 82, 84, 86, 87, 91, 92, 103, 111, 118, 120, 131, 132, 133, 134, 137, 139, 140, 141, 142, 143, 148, 151, 152, 183, 219

CADC correlation analog to digital converter 37, 38, 42, 49, 51, 57, 58, 60, 61, 62, 63, 64, 65, 66, 151, 155, 156, 182

CL1 **c**olumnar **l**ower division of the central body, type 1 124, 127, 131, 132, 139

CMOS complementary metal oxide semiconductor 32, 33, 35

CPU central processing unit 29, 31, 34

CPU1 **c**olumnar **p**rotocerebral bridge/**u**pper division of the central body, type 1 125, 126, 127, 128, 132, 135, 136, 139, 140, 141, 147, 183

CPU4 **c**olumnar **p**rotocerebral bridge/**u**pper division of the central body, type 4 124, 125, 127, 128, 129, 131, 132, 133, 134, 137, 138, 139, 140, 141, 142, 143, 145, 147, 150, 183

DAC digital-to-analog converter 39, 47, 52, 54, 85, 103, 134

DMA direct memory access 106

EEPROM electrically erasable programmable read-only memory 55

flop s^{-1} floating point operations per second 30, 31

Flyspi versatile FPGA board with 256 MB RAM and USB 44, 45, 46, 47, 48, 79

FPGA field-programmable gate array 44, 46, 47, 48, 52, 53, 54, 55, 79, 139, 142

GPU graphics processing unit 29, 31, 32

HBP Human Brain Project 3, 6, 38, 42, 43, 44

HDD hard-disk drive 78, 87, 101, 102, 118, 182

HICANN high input count analog neural network 36, 52

HICANN-X high input count analog neural network - X 37, 38, 42, 44, 48, 49, 50, 51, 52, 53, 54, 55, 56, 60, 61, 63, 150, 182

HICANN-DLS high input count analog neural network with digital learning system 37, 38, 39, 41, 42, 43, 44, 45, 46, 47, 48, 49, 51, 52, 54, 57, 63, 156, 182

HPC high performance computing 31, 32

I^{2}C inter-integrated circuit 55, 85, 90

IC integrated circuit 47, 54, 55, 80, 85

IDE integrated development environment 90

JTAG serial programming and debugging interface 44, 45

LCD liquid crystal display 83

LED light-emitting diode 54, 55

LIF leaky integrate-and-fire 18, 19, 20, 21, 22, 23, 30, 31, 34, 40, 41, 118, 132, 133, 134, 135, 181, 182

LSB least significant bit 58, 62, 65, 104

M motor 140, 147

MADC membrane analog to digital converter 43, 44, 46, 51, 52, 66

MNIST Modified National Institute of Standards and Technology, commonly used to refer to a popular dataset of handwritten digits 63, 64

MOSFET metal – oxide – semiconductor-based field-effect transistor 33

OTA operational transconductance amplifier 40, 41

PCB printed circuit board 44, 45, 48, 52, 53, 56, 77, 82, 84, 86, 87, 88, 102, 103, 104, 105, 106, 112

PGA programmable gain amplifier 103, 104, 111

PLL phase-locked loop 44, 51, 52

PPU plasticity processing unit 37, 38, 42, 44, 50, 51, 57, 63, 120, 134, 135, 139, 140, 142, 150, 155

PSP postsynaptic potential 23, 24

Python programming language 108, 109, 134

RAM random access memory 108

SAR successive approximation register 43, 65, 66

SIMD single instruction multiple data 37

SODIMM Small-Outline Dual Inline Memory Module 45, 54

SPI serial peripheral interface 55

SRAM static random-access memory 39, 59

STDP spike-timing dependent plasticity 25, 30, 37, 57, 63, 182

STP short term plasticity 24, 30, 39, 49

TB1 **t**angential protocerebral **b**ridge, type 1 124, 125, 127, 131, 132, 133, 139, 140, 141, 142, 147

TL **t**angential **l**ower division of the central body 124, 127, 129, 131, 139

TN **t**angential **n**oduli 123, 124, 127, 128, 129, 130, 131, 133, 134, 139, 140, 141, 142, 147

VLSI very-large-scale integration 33

7.1 List of Figures

1.1 Cybernetic loop 2
1.2 'Human brain' vs. 'insect brain' in literature 6

2.1 First illustration by Ramon y Cajal 12
2.2 The neuron 13
2.3 Real and calculated action potentials 15
2.4 Circuit diagram of the Hodgkin-Huxley model 16
2.5 Equivalent circuit diagram of the LIF equations 18
2.6 Constructed synaptic input current 19
2.7 Response of the LIF equations with a non-zero refractory period . . 20

2.8 Rate response of an LIF neuron to constant stimulus 21
2.9 Rate response of an LIF neuron to a noisy stimulus 21
2.10 Constructed synaptic input current 24
2.11 STDP 25

3.1 Simplified ANNCORE architecture 38
3.2 HICANN-DLSv2 layout 39
3.3 Block schematic of the HICANN-DLSv2 neuron 41
3.4 HICANN-DLSv3 chip on the chip carrier 43
3.5 Block-level diagram of the HICANN-DLSv3 setup 45
3.6 Block-level schematic of the HICANN-DLSv3 baseboard 46
3.7 Photo of the HICANN-DLSv3 baseboard 48
3.8 HICANN-X chip on its chip carrier 49
3.9 HICANN-X block-level schematic 50
3.10 HICANN-X block-level quadrant 50
3.11 Cube setup with xBoard 53
3.12 Block-level schematic of the xBoard 54
3.13 HICANN-X chip carrier board 55
3.14 A few HICANN-X test setups 56
3.15 CADC architecture 57
3.16 CADC column 59
3.17 Uncalibrated and calibrated CADC response functions 60
3.18 Uncalibrated and calibrated CADC deviations 61
3.19 Conversion energy for different ADCs 62
3.20 Correlation recordings of 31 synapses 63
3.21 Parallel recording of membrane traces with the CADC 64

4.1 *Drosophila melanogaster*'s physical reactions 69
4.2 Schematic of biological motor units 71
4.3 Force response of motor units 71
4.4 Human retina 72
4.5 The spatial representation of visual information in the striate cortex 74
4.6 Spike response of mechanoreceptive sensory neurons 75
4.7 HDD with magnetic storage disks and voice coil actuator 78
4.8 Voice coil within the magnetic field 79
4.9 The PlayPen 80
4.10 Pulse shaper of the PlayPen prototype 81
4.11 Sensor arrangements with one, three, and seven channels 82
4.12 Photo diode within its aperture cabinet 82
4.13 Convolution of the receptive field 83
4.14 PlayPen prototype 86
4.15 Voltage to spike converters of the PlayPen prototype 89
4.16 Block schematic of a signal chain 90
4.17 Light edge experiment schematic 91
4.18 Block schematic of the edge tracing experiment 92
4.19 Experimental setup for a neuromorphic maximum finder 93

4.20 Neural network for maximum brightness tracing 97
4.21 Position of the sensor head and distance to the center of maximum brightness 99
4.22 Sensor input and motor outputs during a regulation cycle 100
4.23 Sensor input and motor outputs while the location of maximum brightness is moving 100
4.24 Mechanical response of the PP2 sensor head 102
4.25 Block schematic of the PlayPen-2 signal path 103
4.26 Cabinet of the PlayPen-2 sensor head 105
4.27 PlayPen-2 sensor head 105
4.28 PlayPen-2 rotary sensor 106
4.29 The complete PlayPen-2 setup 110
4.30 Experiment schematic and data flow for maximum brightness tracing 111
4.31 Detailed data flow and network architecture for brightness maximization 112
4.32 Trajectories of the sensor head starting from four different positions 113
4.33 Position and optical sensor signals over time 113
4.34 Evolution of the maximum finder starting from a heuristically chosen initial weight configuration 115
4.35 Evolution of the maximum finder starting from a random initial weight configuration 117

5.1 *Central complex* of *Megalopta genalis* 121
5.2 Brain of the tropical nocturnal bee *Megalopta genalis* 121
5.3 Path integration in the Saharan dessert ant *Cataglyphis fortis* . . . 122
5.4 Orientation encoding in the central complex 124
5.5 CPU1 neurons in the brain of the monarch butterfly 126
5.6 Abstract point neuron representation of the path integration network 128
5.7 Coordinate system of the insectoid agent 129
5.8 Axoaxonic depression and facilitation 133
5.9 Output response of the calibrated CPU1 neurons 136
5.10 Deviation of the output responses of the calibrated CPU1 neurons . 136
5.11 Output response of the CPU4 neurons 138
5.12 Network architecture of the BrainScaleS-2 implementation 141
5.13 Experiment schedule 143
5.14 Network activity and insect trajectory 144
5.15 Effect of h and k on the network performance 145
5.16 Effect of h and k on the network performance (heatmaps) 146
5.17 Statistical performance and evolutionary optimization 149

A.1 Network activity and trajectory for $h = 0.050$ and $k = -6$. 187
A.2 Network activity and trajectory for $h = 0.022$ and $k = -6$. 188
A.3 Network activity and trajectory for $h = 0.022$ and $k = 0$. 189
A.4 Network activity and trajectory for $h = 0.034$ and $k = 2$. 190
A.5 Network activity and trajectory for $h = 0.050$ and $k = 6$. 191
A.6 Network activity and trajectory for $h = 0.022$ and $k = 6$. 192

7.2 List of Tables

4.1 Amplifier output states and voice coil responses 81
4.2 Translation of arm rotations into directions of sensor head movements 93
4.3 State matrix of the maximum finder 96

5.1 Statistical insect performance 148

Part I

Appendix

A Additional figures

A.1 Insect trajectories

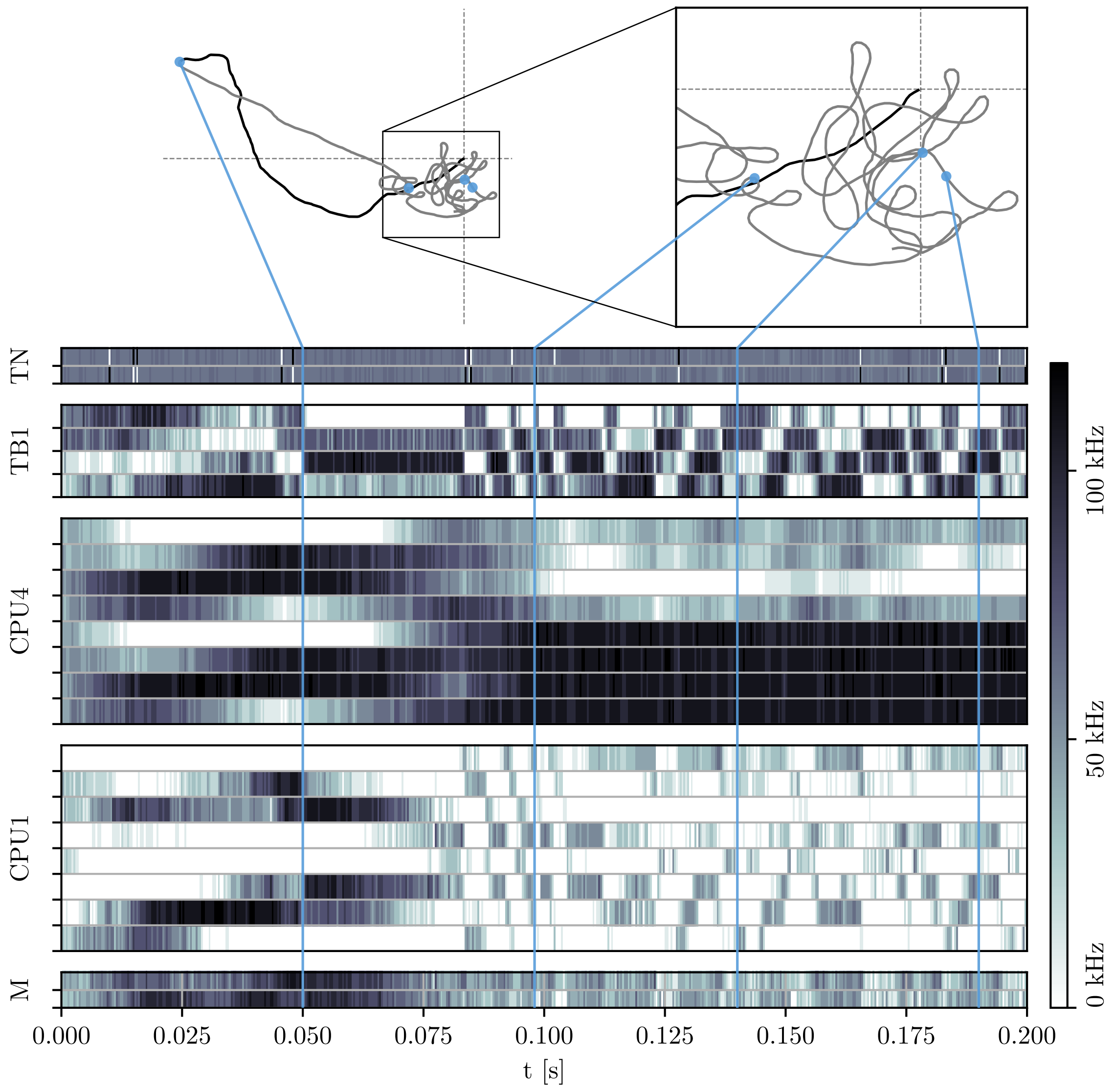


Figure A.1: Network activity and trajectory for $h = 0.050$ and $k = -6$. See the description of figure 5.14 for a detailed description.

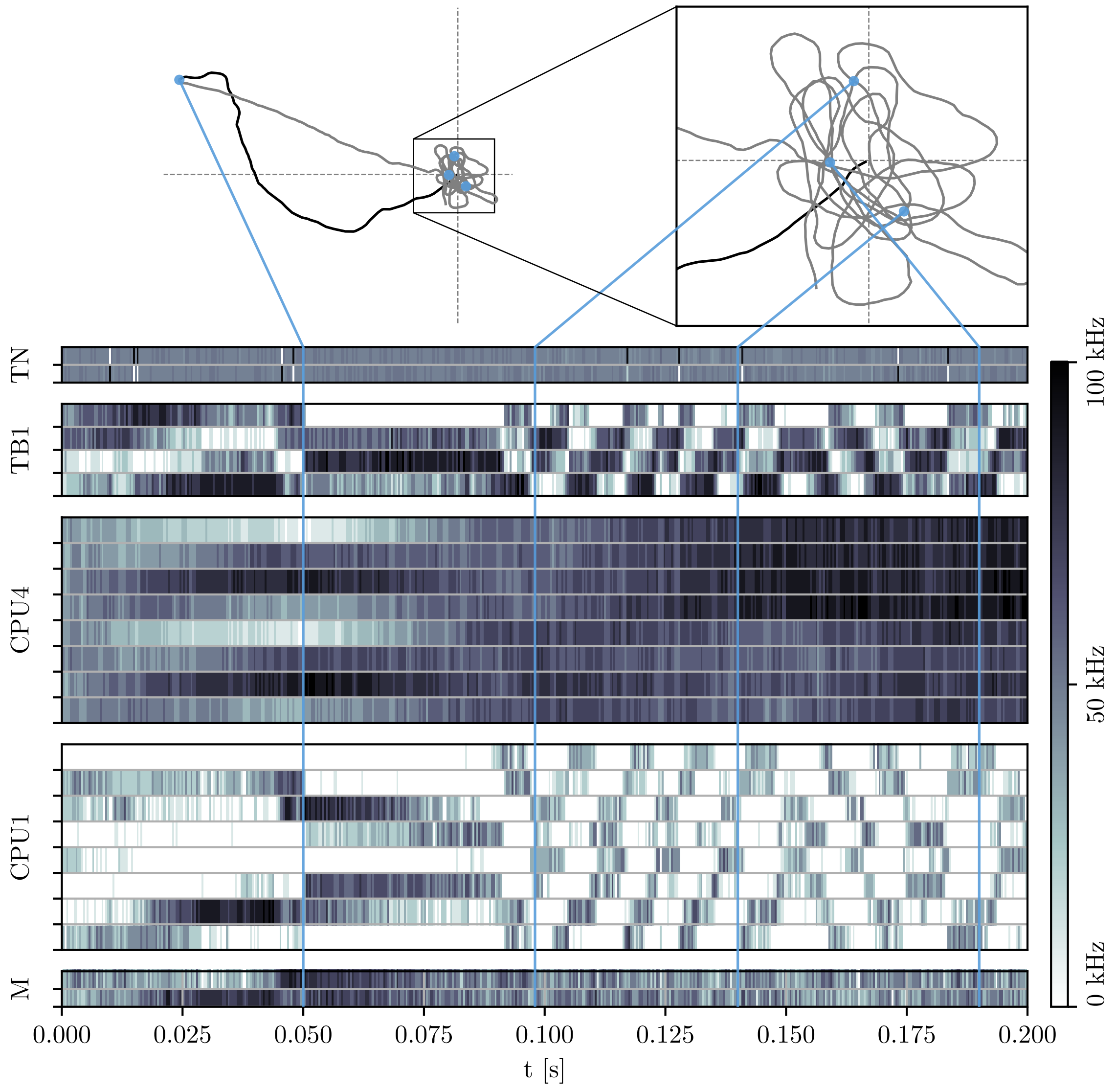


Figure A.2: Network activity and trajectory for $h = 0.022$ and $k = -6$. See the description of figure 5.14 for a detailed description.

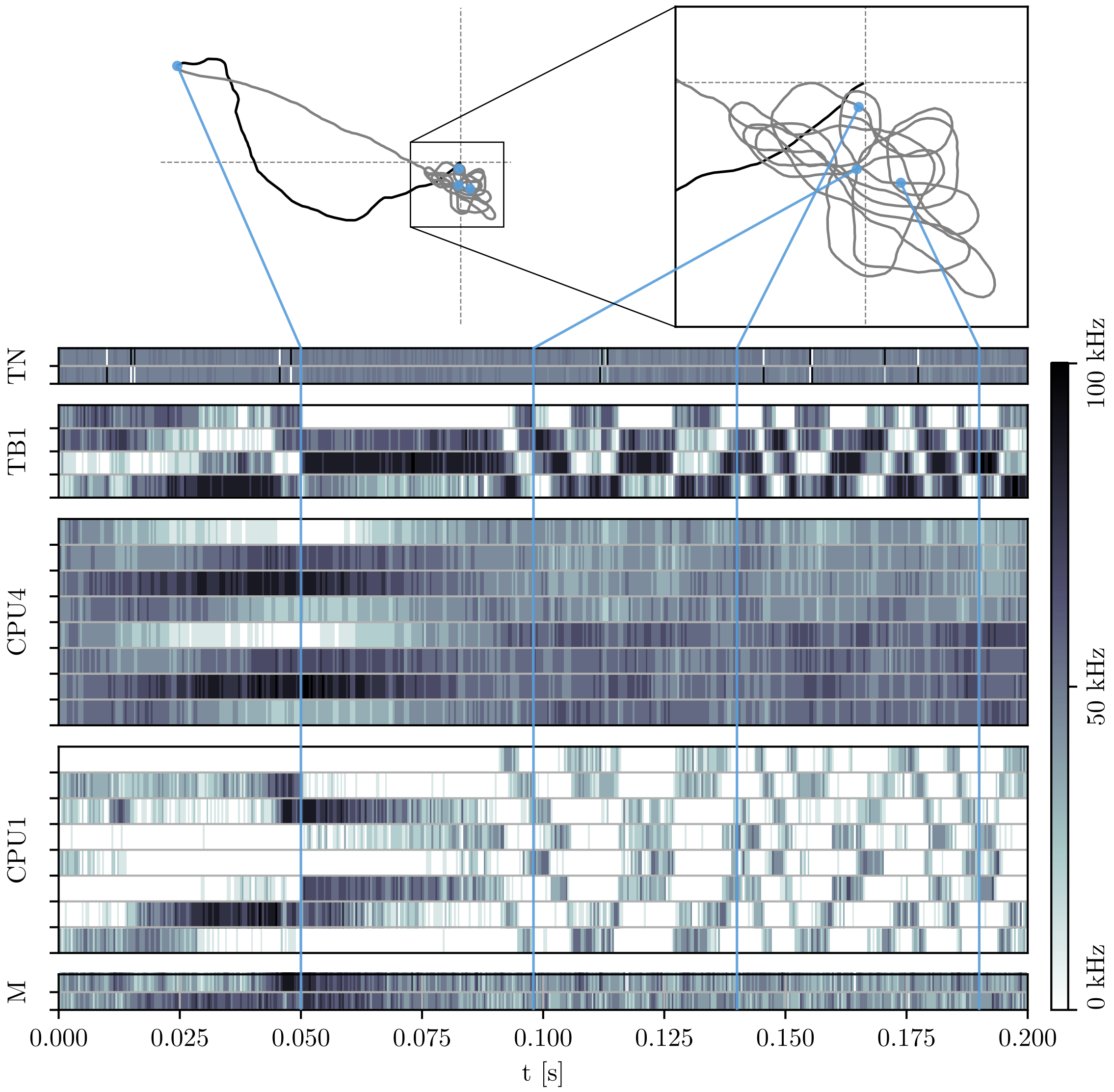


Figure A.3: Network activity and trajectory for $h = 0.022$ and $k = 0$. See the description of figure 5.14 for a detailed description.

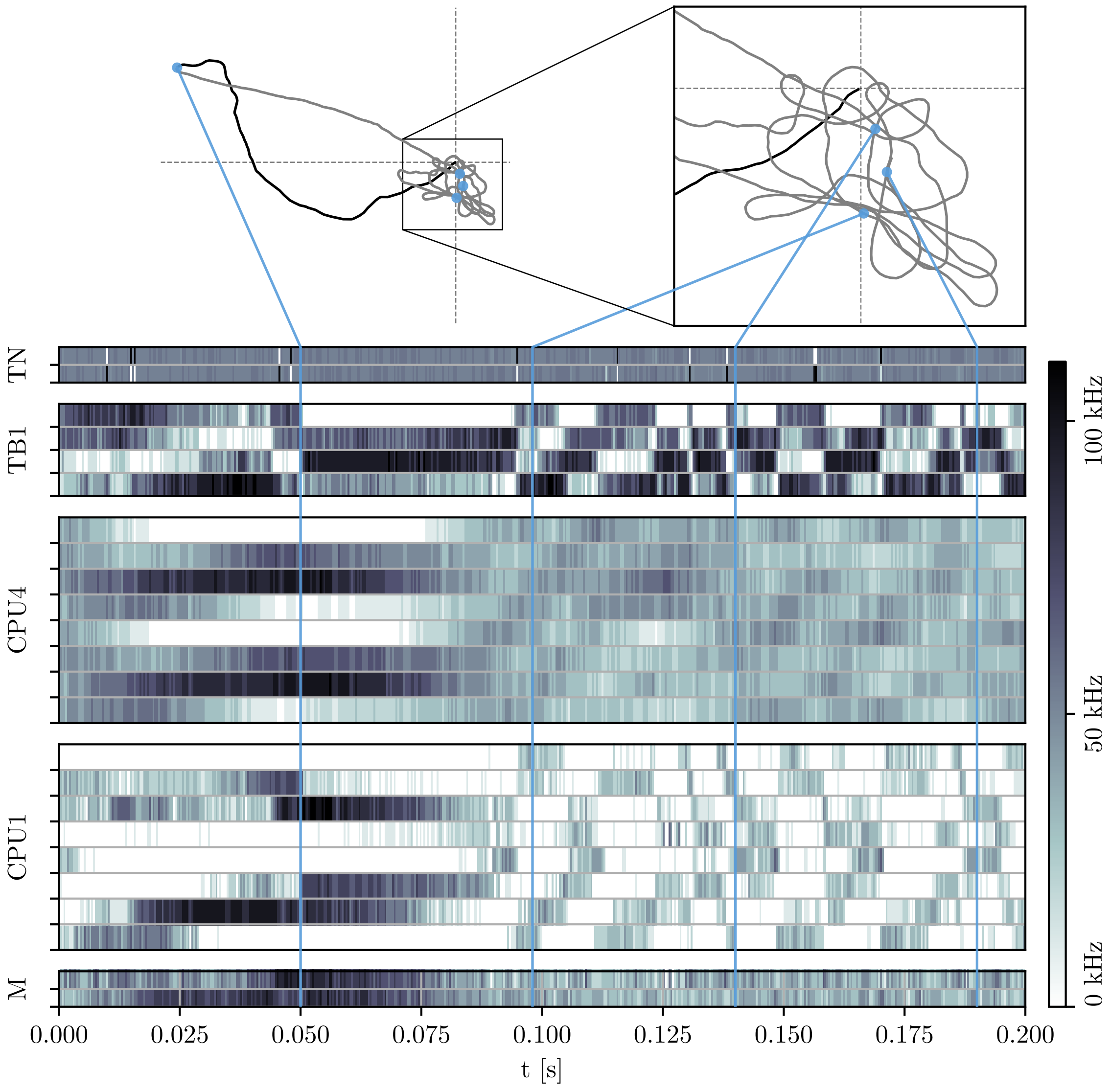


Figure A.4: Network activity and trajectory for $h = 0.034$ and $k = 2$. See the description of figure 5.14 for a detailed description.

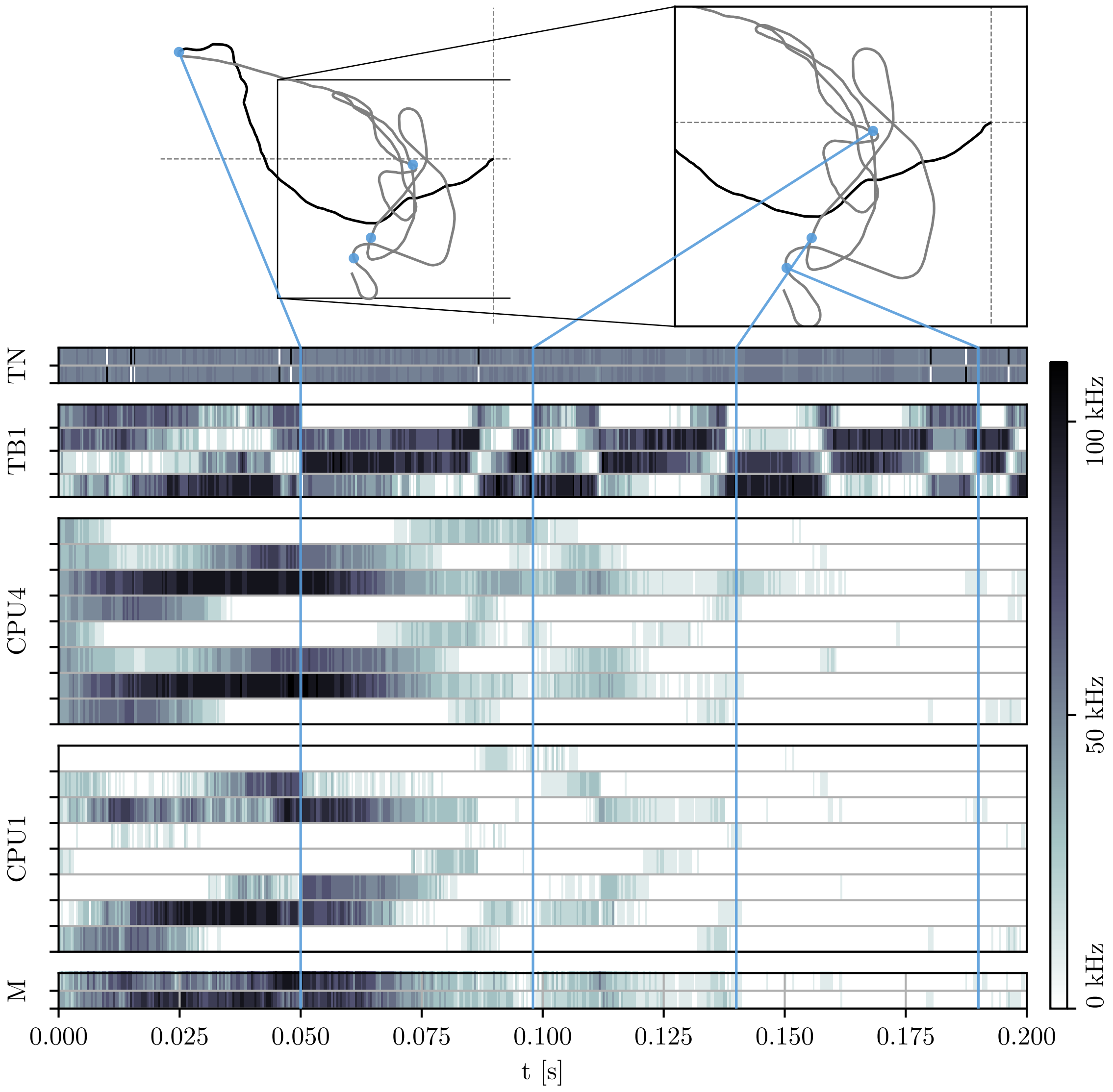


Figure A.5: Network activity and trajectory for $h = 0.050$ and $k = 6$. See the description of figure 5.14 for a detailed description.

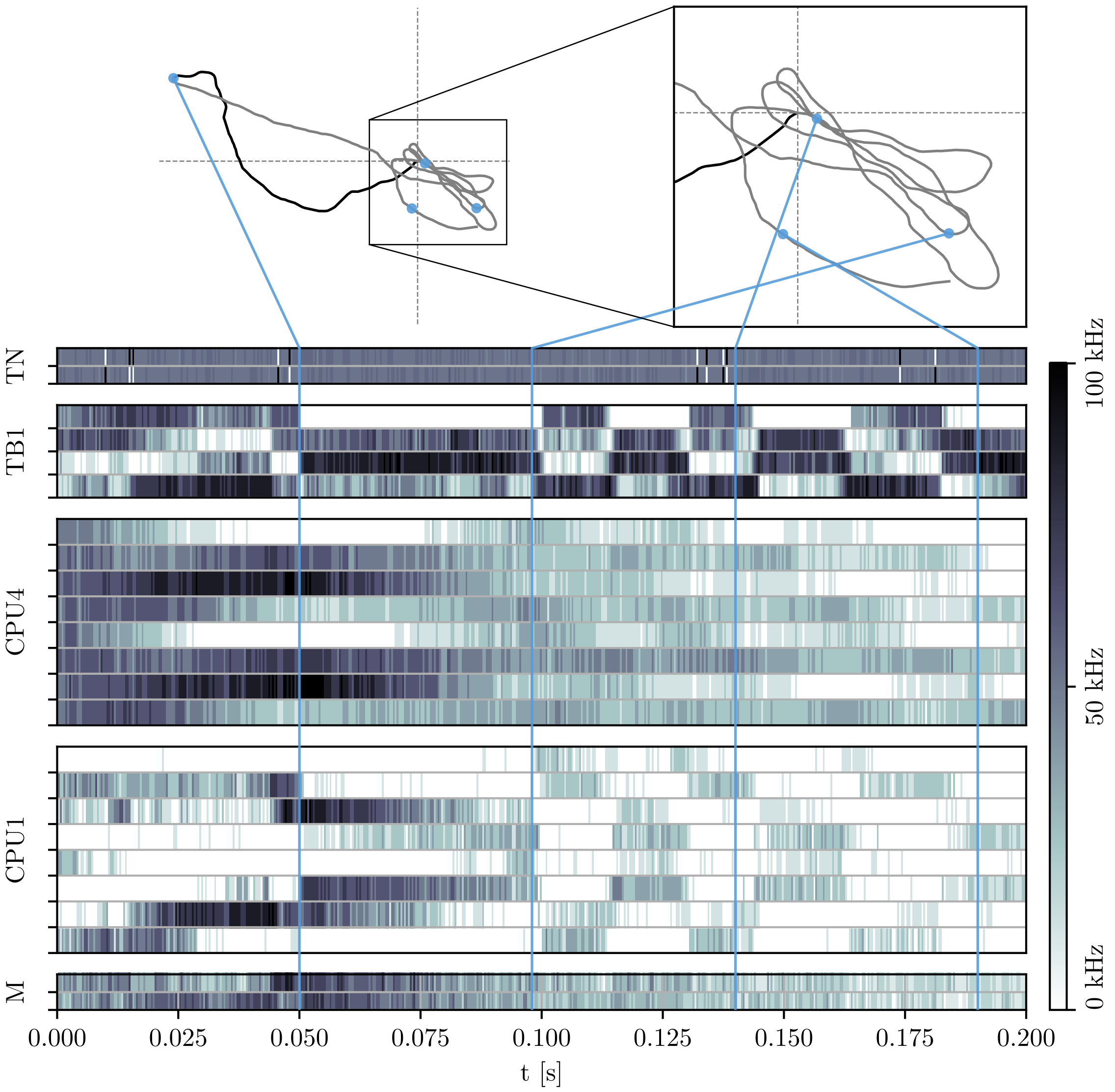


Figure A.6: Network activity and trajectory for $h = 0.022$ and $k = 6$. See the description of figure 5.14 for a detailed description.

B Code snippets

B.1 CPU4 read access

The supersynaptic weights are read out with the following ‘C++‘ routine.

```
void InsectBrain::readCPU4Weights(void)
{
    register vector uint8_t tmp;
    register vector uint8_t tmp2;
    register vector uint8_t read_weights;

    vector uint8_t accumulator_msb;
    vector uint8_t accumulator_lsb;
    vector uint8_t zeros;

    for(uint8_t i = 0; i < 16; i++) zeros[i] = 0;

    // Set accumulator to 0
    asm volatile (
        // Set accumulator to 0.
        // Load zeros to tmp:
        "fxvlax %[tmp], 0, %[zeros]\n"
        // Move content of tmp to accumulator, aligned to the right:
        "fxvmtacb %[tmp]\n"
        "sync\n"
        : [tmp] "=&kv" (tmp)
        : [zeros] "r" (&zeros)
        :
    );

    // Accumulate weights additively
    for(uint8_t row = 0; row < INSECT_CPU4_SYNAPSE_DEPTH; row++){
        uint8_t idx = 2*row;
        asm volatile (
            // Read synaptic weights
            // Load weight slice to read_weights:
            "fxvinx %[read_weights], %[dls_weight_base], %[idx]\n"
            // Add weights to double precision accumulator in a saturating manner:
            "fxvaddactacbm %[read_weights]\n"
            "sync\n"
            : [read_weights] "=&kv" (read_weights)
            : [dls_weight_base] "b" (dls_weight_base),
              [idx] "r" (idx)
            :
        );
    }
    // Save most significant bytes
    asm volatile (
        // Load zeros to tmp:
        "fxvlax %[tmp], 0, %[zeros]\n"
        // Add zeros to ACC and store in tmp2
        "fxvaddacbfs %[tmp2], %[tmp]\n"
        // Store contents of tmp2 to address starting at msb_addr
        "fxvstax %[tmp2], 0, %[msb_addr]\n"
        "sync\n"
        : [tmp] "=&kv" (tmp),
          [tmp2] "=&kv" (tmp2)
        : [zeros] "r" (&zeros),
          [msb_addr] "r" (&accumulator_msb)
        :
    );

    // Save least significant bytes
    asm volatile (
        // Add accumulator byte modulo only to get the accumulator LSB.
        // Load zeros to tmp:
        "fxvlax %[tmp], 0, %[zeros]\n"
        // Add contents of tmp to ACC and write truncated result to tmp
        "fxvaddacbm %[tmp2], %[tmp]\n"
        // Store contents of tmp to address starting at msb_addr
        "fxvstax %[tmp2], 0, %[lsb_addr]\n"
        "sync\n"
        : [tmp] "=&kv" (tmp),
          [tmp2] "=&kv" (tmp2)
        : [zeros] "r" (&zeros),
          [lsb_addr] "r" (&accumulator_lsb)
        :
    );

    // Write weights to the globally accessible memory locations
    for(uint8_t col = 0; col < 2*INSECT_BRAIN_ANGULAR_RESOLUTION; col++) {
        weights[col] = accumulator_lsb[INSECT_BRAIN_CPU4_NEURON_IDX[col]];
        weights[col] |= (accumulator_msb[INSECT_BRAIN_CPU4_NEURON_IDX[col]] << 8);
    }
}
```

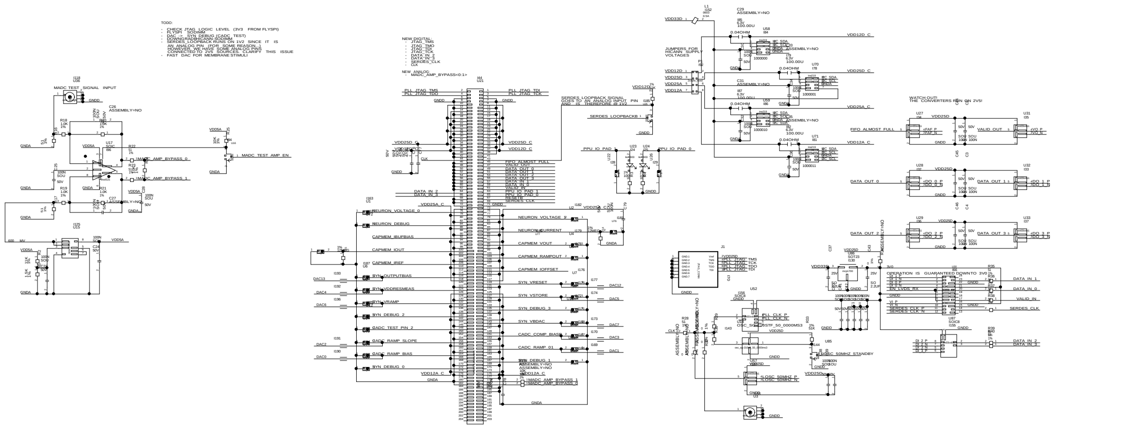
B.2 CPU4 write access

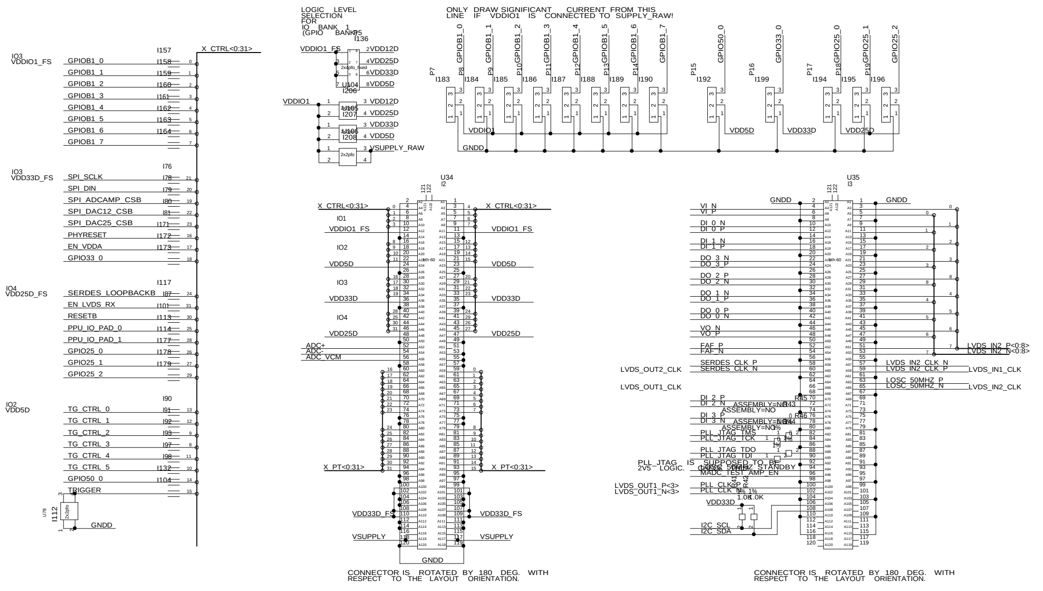
The supersynaptic weights are written with the following ‘C++‘ routine.

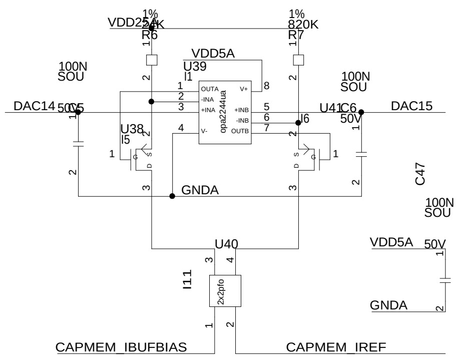
```
void InsectBrain::writeCPU4Weights(void)
{
    vector uint8_t zeros = vec_splat_u8(0);
    vector uint8_t ones = vec_splat_u8(1);
    vector uint8_t sixtythree = vec_splat_u8(63);
    volatile vector uint8_t weights_to_fill;
    volatile vector uint8_t weights_residue;

    // This loop takes ~2.2us
    for (uint8_t i = 0; i < 2*INSECT_BRAIN_ANGULAR_RESOLUTION; i++) {
        uint8_t to_fill = weights[i]/63;
        weights_to_fill[INSECT_BRAIN_CPU4_NEURON_IDX[i]] = to_fill;
        weights_residue[INSECT_BRAIN_CPU4_NEURON_IDX[i]] = weights[i] - 63*to_fill;
    }

    // Conditions:
    // 0: always
    // 1: greater then
    // 2: less then
    // 3: equals
    for(uint8_t row = 0; row < INSECT_CPU4_SYNAPSE_DEPTH; row++){
        uint8_t idx = 2*row;
        // clang-format off
        asm volatile (
            // Load weights_to_fill to tmp1:
            // "fxvlax %[tmp1], 0, %[fill_addr]\n"
            "fxvlax 1, 0, %[fill_addr]\n"
            // Splatter idx to tmp2
            // "fxvsplatb %[tmp2], %[row]\n"
            "fxvsplatb 2, %[row]\n"
            // Subtract idx from weights_to_fill and store the difference in tmp1
            // "fxvsubbm %[tmp1], %[tmp1], %[tmp2]\n"
            "fxvsubbm 1, 1, 2\n"
            // Compare this difference to 0:
            // "fxvcmpb %[tmp1]\n"
            "fxvcmpb 1\n"
            // Select either 63 or 0 to write to synapse
            // If difference > 0: write 63, else write 0
            // "fxvsel %[tmp2], %[zeros], %[sixtythree], 1\n"
            "fxvsel 2, %[zeros], %[sixtythree], 1\n"

            // Load residues to tmp3
            // "fxvlax %[tmp3], 0, %[residue_addr]\n"
            "fxvlax 3, 0, %[residue_addr]\n"
            // Add 1 to the difference and compare again:
            // "fxvcmpb %[tmp1]\n"
            "fxvcmpb 1\n"
            // Select residue, if the index is correct
            // "fxvsel %[tmp2], %[tmp2], %[tmp3], 3\n"
            "fxvsel 2, 2, 3, 3\n"
            // Write the weights
            // "fxvoutx %[tmp2], %[dls_weight_base], %[idx]\n"
            "fxvoutx 2, %[dls_weight_base], %[idx]\n"

            :
            // [tmp1] "=&kv" (tmp1),
            // [tmp2] "=&kv" (tmp2),
            // [tmp3] "=&kv" (tmp3)
            : [fill_addr] "r" (&weights_to_fill),
              [residue_addr] "r" (&weights_residue),
              [sixtythree] "kv" (sixtythree),
              [zeros] "kv" (zeros),
              [ones] "kv" (ones),
              [dls_weight_base] "b" (dls_weight_base),
              [idx] "r" (idx),
              [row] "r" (row)
            :
        );
        // clang-format on
    }
}
```

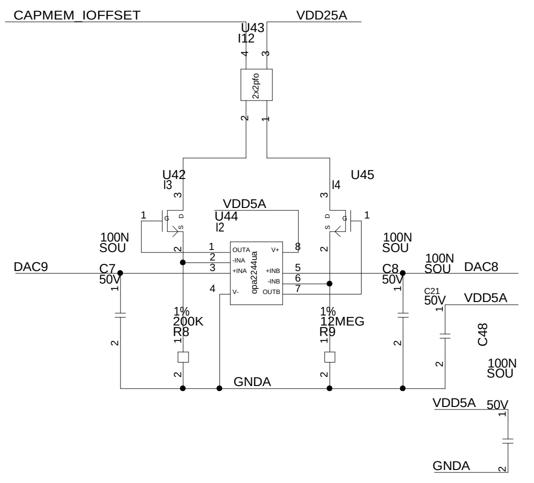
C Schematics

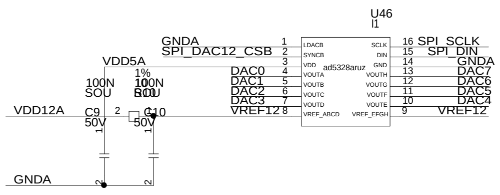
C.1 Baseboard for the 3rd generation prototype

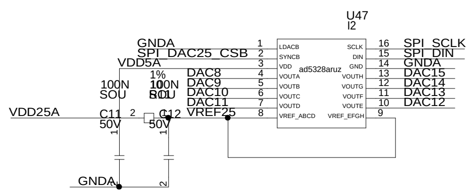


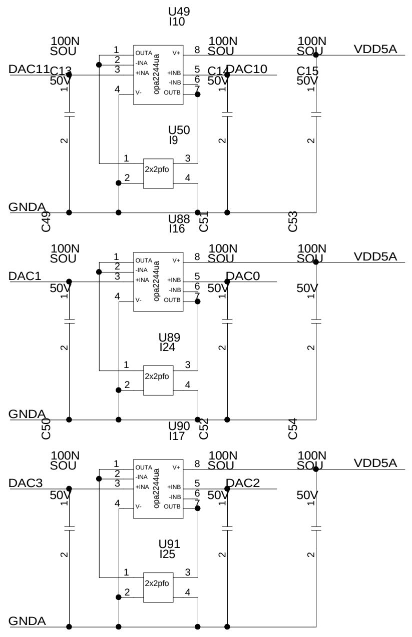


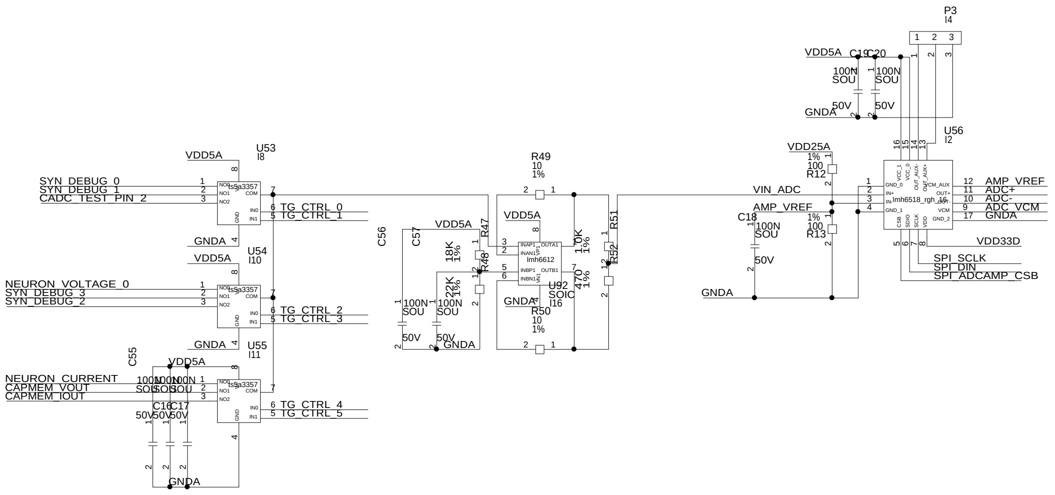

VDD25A
1%
24K
R6
1%
820K
R7
VDD5A
U39
I1
opa2244ua
100N
SOU
DAC14
C5
50V
U38
I5
U41
I6
C6
50V
DAC15
C47
GNDA
U40
I11
2x2pfo
VDD5A
50V
GNDA
CAPMEM_IBUFBIAS
CAPMEM_IREF

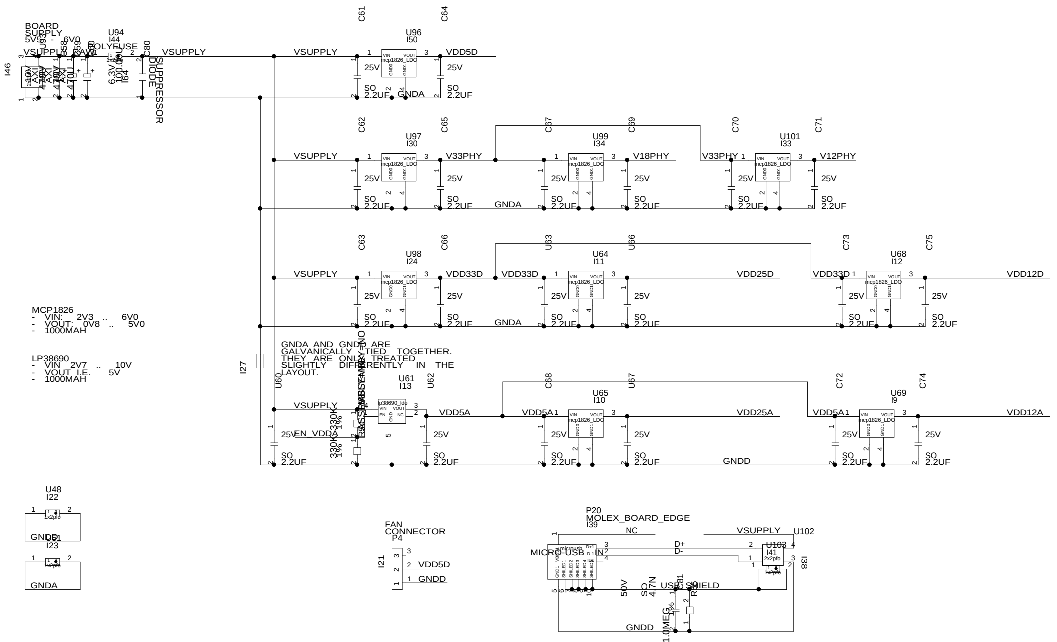

CAPMEM_IOFFSET
VDD25A
U43
I12
2x2pfo
U42
I3
U45
I4
VDD5A
U44
I2
opa2244ua
100N
SOU
DAC9
C7
50V
C8
50V
DAC8
C21
50V
VDD5A
C48
1%
200K
R8
1%
12MEG
R9
GNDA
VDD5A
50V
GNDA

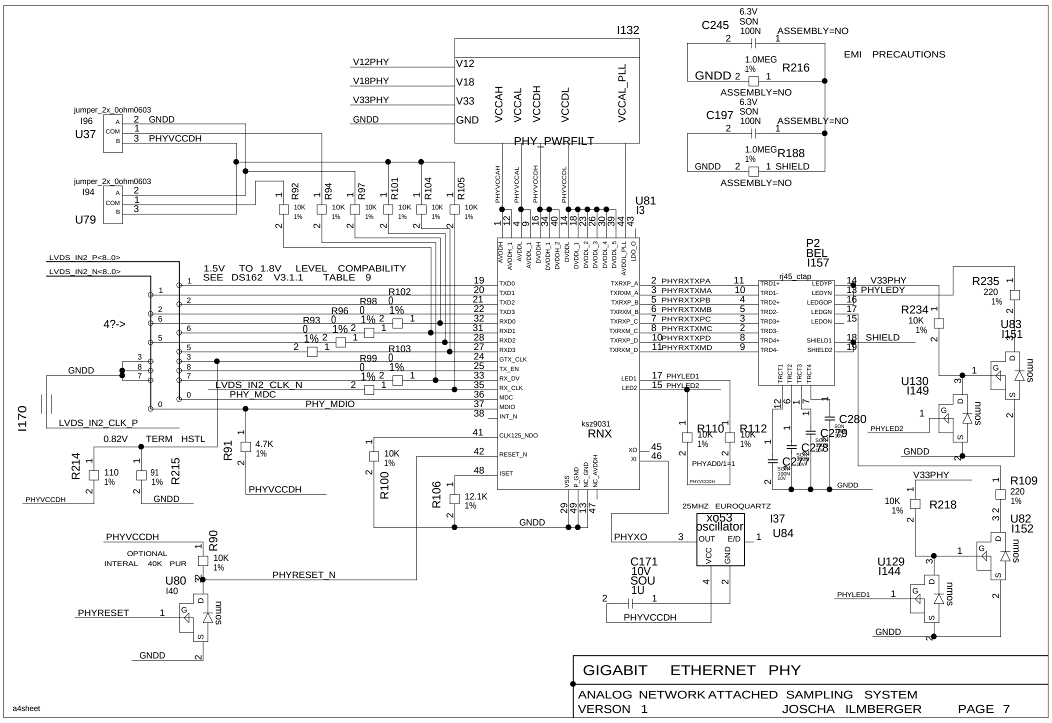

U46
I1
ad5328aruz
GNDA
SPI_DAC12_CSB
VDD5A
DAC0
DAC1
DAC2
DAC3
VREF12
SPI_SCLK
SPI_DIN
GNDA
DAC7
DAC6
DAC5
DAC4
VREF12
1%
100N
SOU
VDD12A
C9
C10
50V
GNDA


U47
I2
ad5328aruz
GNDA
SPI_DAC25_CSB
VDD5A
DAC8
DAC9
DAC10
DAC11
VREF25
SPI_SCLK
SPI_DIN
GNDA
DAC15
DAC14
DAC13
DAC12
1%
100N
SOU
VDD25A
C11
C12
50V
GNDA


U49
I10
opa2244ua
100N
SOU
VDD5A
DAC11
C13
50V
C14
DAC10
C15
U50
I9
2x2pfo
GNDA
C49
C51
C53
U88
I16
DAC1
DAC0
U89
I24
C50
C52
C54
U90
I17
DAC3
DAC2
U91
I25
GNDA

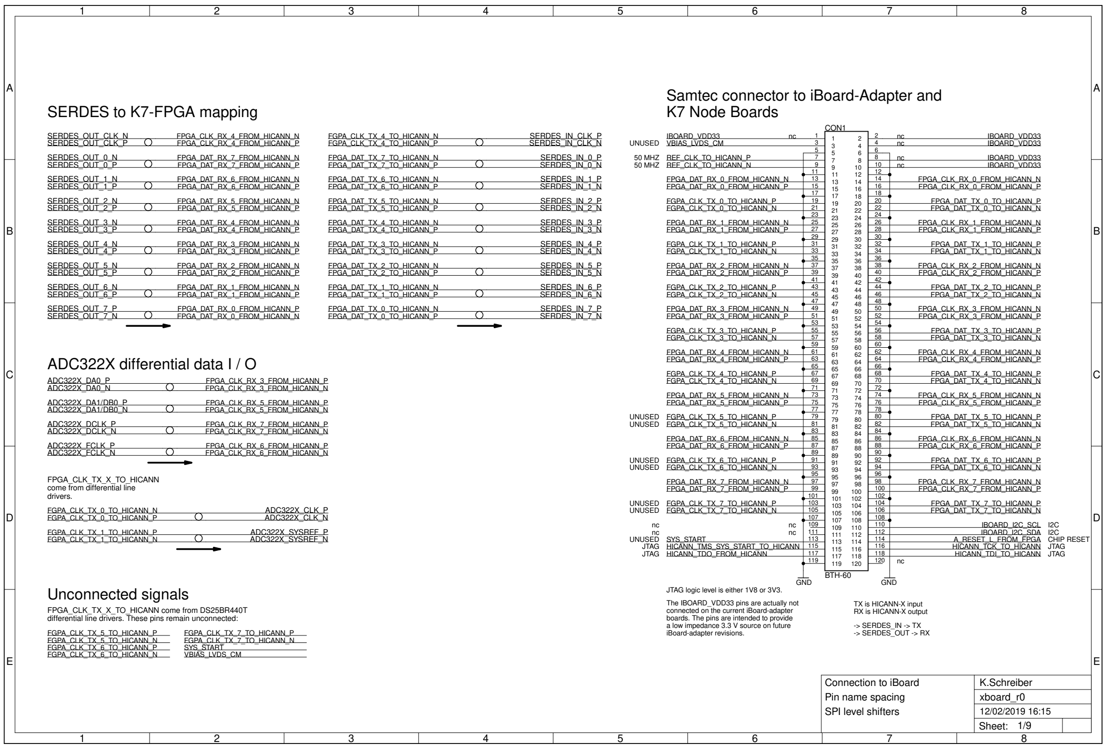

SYN_DEBUG_0
SYN_DEBUG_1
CADC_TEST_PIN_2
NEURON_VOLTAGE_0
SYN_DEBUG_3
SYN_DEBUG_2
NEURON_CURRENT
CAPMEM_VOUT
CAPMEM_IOUT
TG_CTRL_0
TG_CTRL_1
TG_CTRL_2
TG_CTRL_3
TG_CTRL_4
TG_CTRL_5
VDD5A
GNDA
U53
U54
U55
ts5a3357
lmh6612
U92
SOIC
R49
R50
VDD25A
VIN_ADC
AMP_VREF
U56
lmh6518_rgh
AMP_VREF
ADC+
ADC-
ADC_VCM
GNDA
VDD33D
SPI_SCLK
SPI_DIN
SPI_ADCAMP_CSB
P3

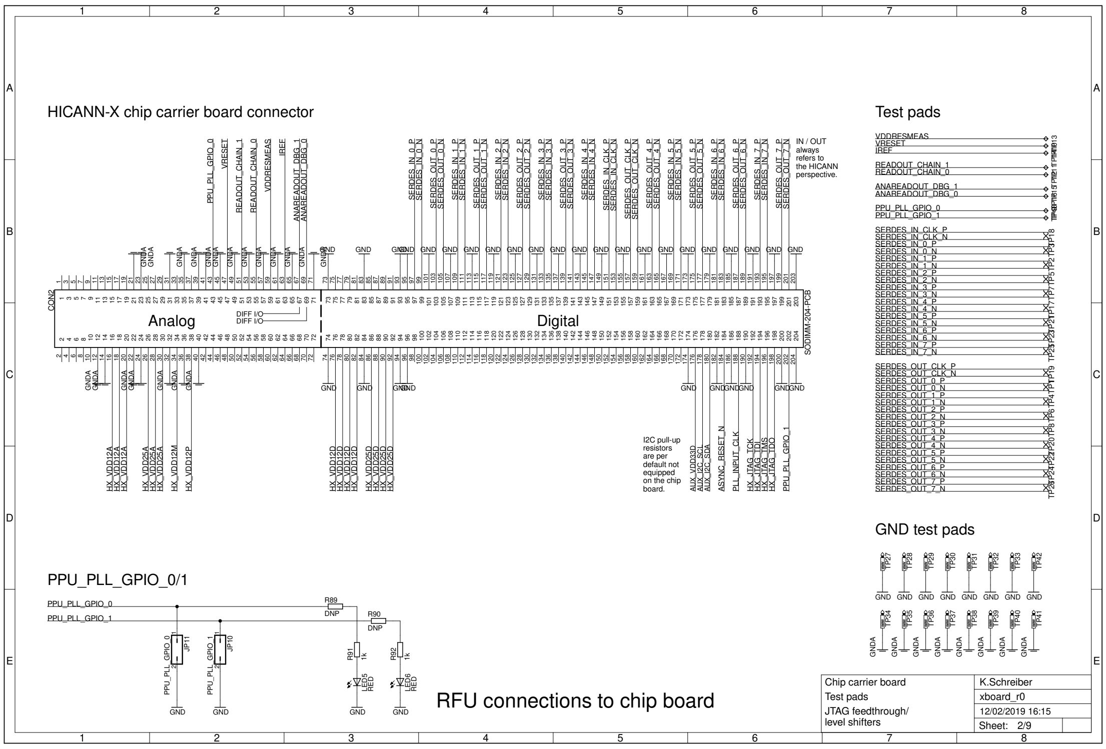

BOARD SUPPLY 5V5 .. 6V0
VSUPPLY_RAW
POLYFUSE
SUPPRESSOR
VSUPPLY
mcp1826_LDO
VDD5D
V33PHY
V18PHY
V12PHY
VDD33D
VDD25D
VDD12D
VDD5A
VDD25A
VDD12A
GNDA
GNDD
MCP1826
- VIN: 2V3 .. 6V0
- VOUT: 0V8 .. 5V0
- 1000MAH
LP38690
- VIN 2V7 .. 10V
- VOUT I.E. 5V
- 1000MAH
GNDA AND GNDD ARE GALVANICALLY TIED TOGETHER. THEY ARE ONLY TREATED SLIGHTLY DIFFERENTLY IN THE LAYOUT.
lp38690_ldo
EN_VDD5A
FAN CONNECTOR
P4
VDD5D
GNDD
P20
MOLEX_BOARD_EDGE
MICRO-USB
NC
D+
D-
USB_SHIELD
U102

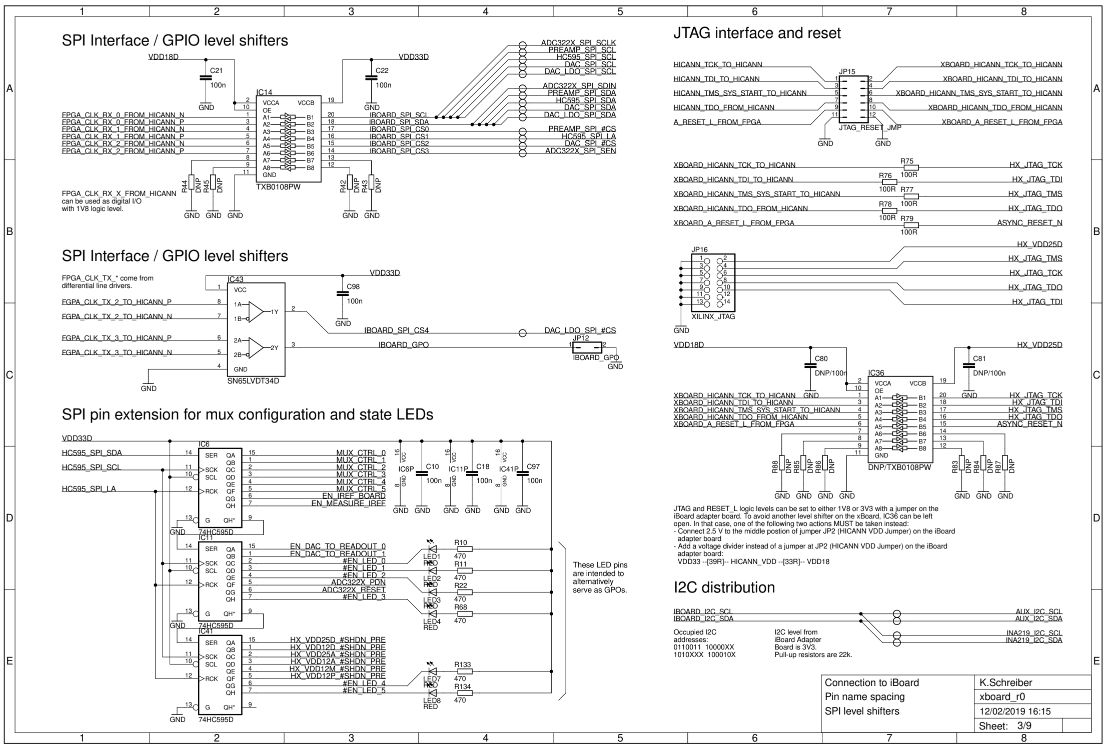


GIGABIT ETHERNET PHY
ANALOG NETWORK ATTACHED SAMPLING SYSTEM
VERSON 1
JOSCHA ILMBERGER
PAGE 7
a4sheet
ksz9031
RNX
U81
I132
PHY_PWRFILT
EMI PRECAUTIONS
ASSEMBLY=NO
1.5V TO 1.8V LEVEL COMPABILITY
SEE DS162 V3.1.1 TABLE 9
rj45_ctap
P2
BEL
I157
25MHZ EUROQUARTZ
xo53
oscillator
U84
I37
OPTIONAL
INTERAL 40K PUR
PHYRESET_N
PHYRESET
LVDS_IN2_CLK_N
LVDS_IN2_CLK_P
PHY_MDC
PHY_MDIO
0.82V TERM HSTL

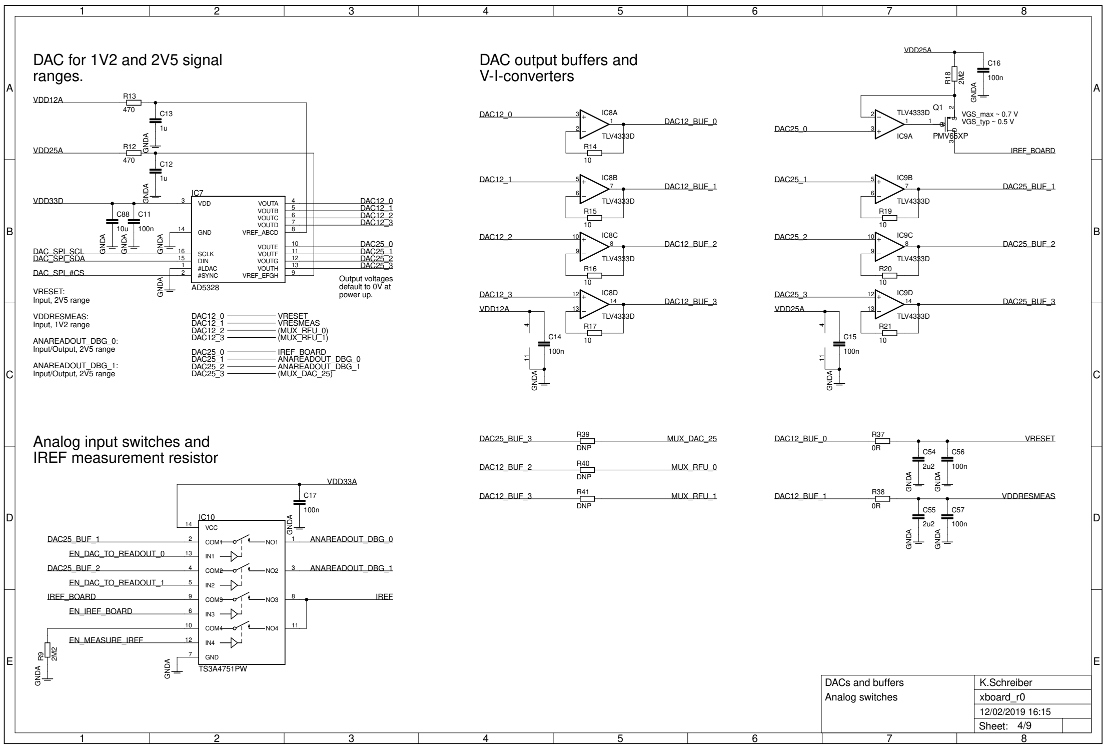
C.2 xBoard


SERDES to K7-FPGA mapping
ADC322X differential data I / O
FPGA_CLK_TX_X_TO_HICANN come from differential line drivers.
Unconnected signals
FPGA_CLK_TX_X_TO_HICANN come from DS25BR440T differential line drivers. These pins remain unconnected:
Samtec connector to iBoard-Adapter and K7 Node Boards
CON1
BTH-60
JTAG logic level is either 1V8 or 3V3.
The IBOARD_VDD33 pins are actually not connected on the current iBoard-adapter boards. The pins are intended to provide a low impedance 3.3 V source on future iBoard-adapter revisions.
TX is HICANN-X input
RX is HICANN-X output
-> SERDES_IN -> TX
-> SERDES_OUT -> RX
Connection to iBoard
Pin name spacing
SPI level shifters
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 1/9

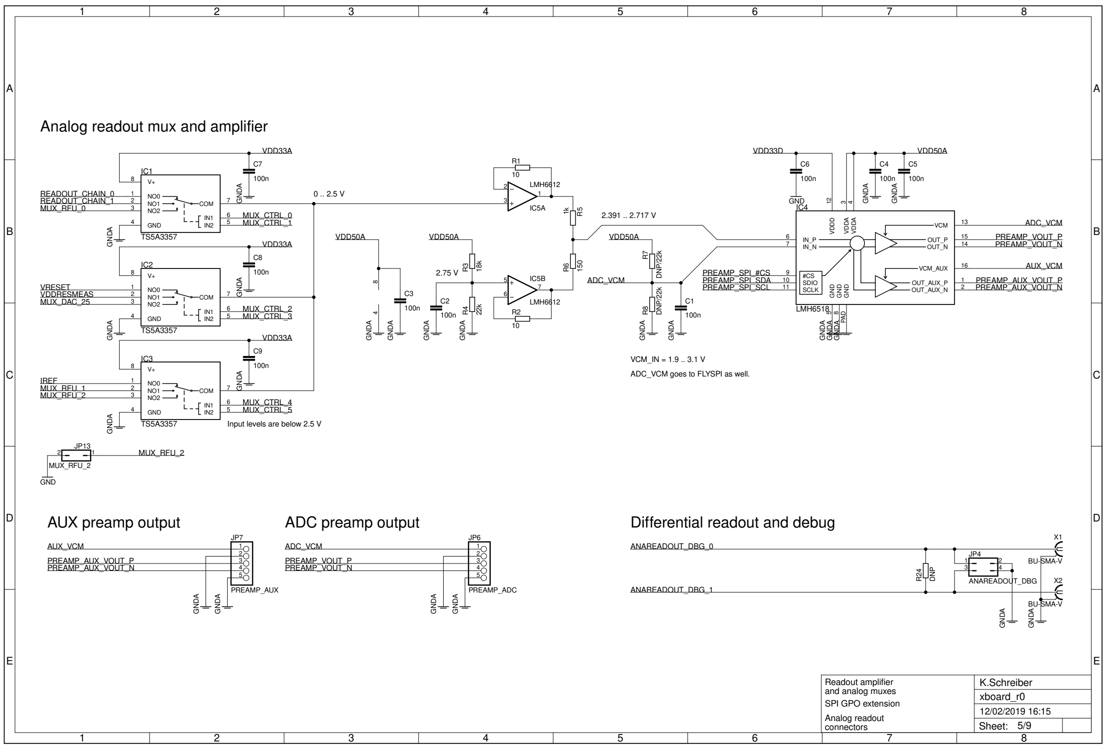

HICANN-X chip carrier board connector
CON2
Analog
Digital
DIFF I/O
SODIMM-204-PCB
IN / OUT always refers to the HICANN perspective.
I2C pull-up resistors are per default not equipped on the chip board.
Test pads
GND test pads
PPU_PLL_GPIO_0/1
PPU_PLL_GPIO_0
PPU_PLL_GPIO_1
R89 DNP
R90 DNP
R91 1k
R92 1k
LED5 RED
LED6 RED
JP11
JP10
RFU connections to chip board
Chip carrier board
Test pads
JTAG feedthrough/ level shifters
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 2/9

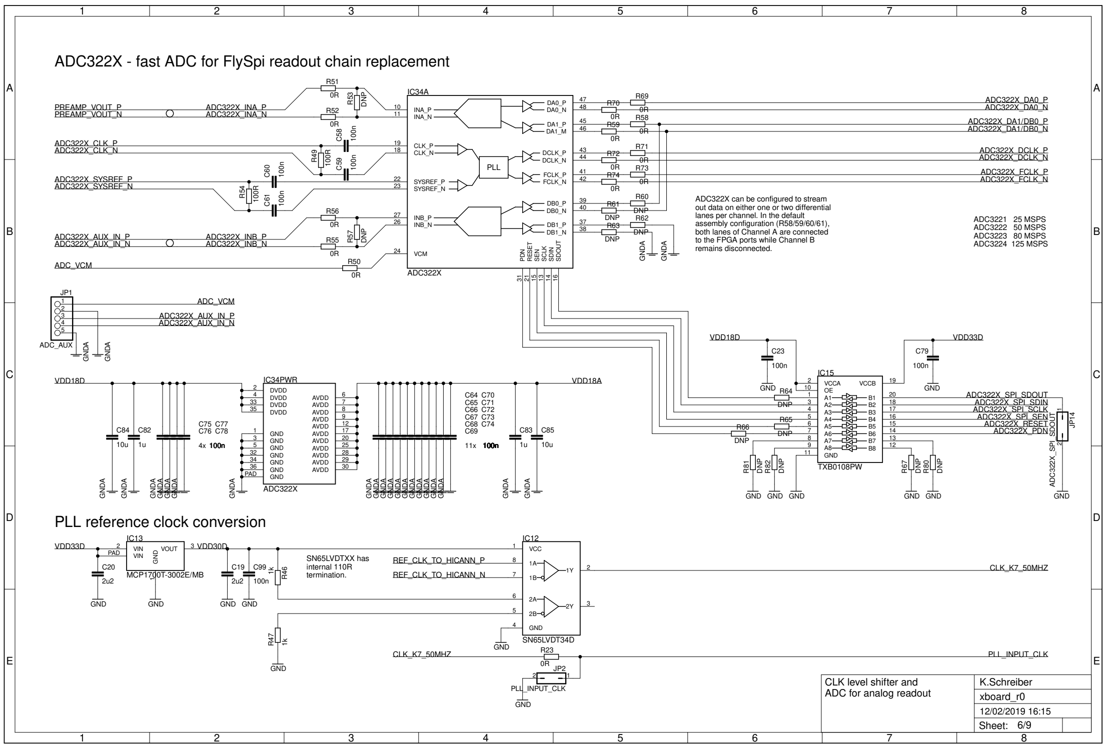


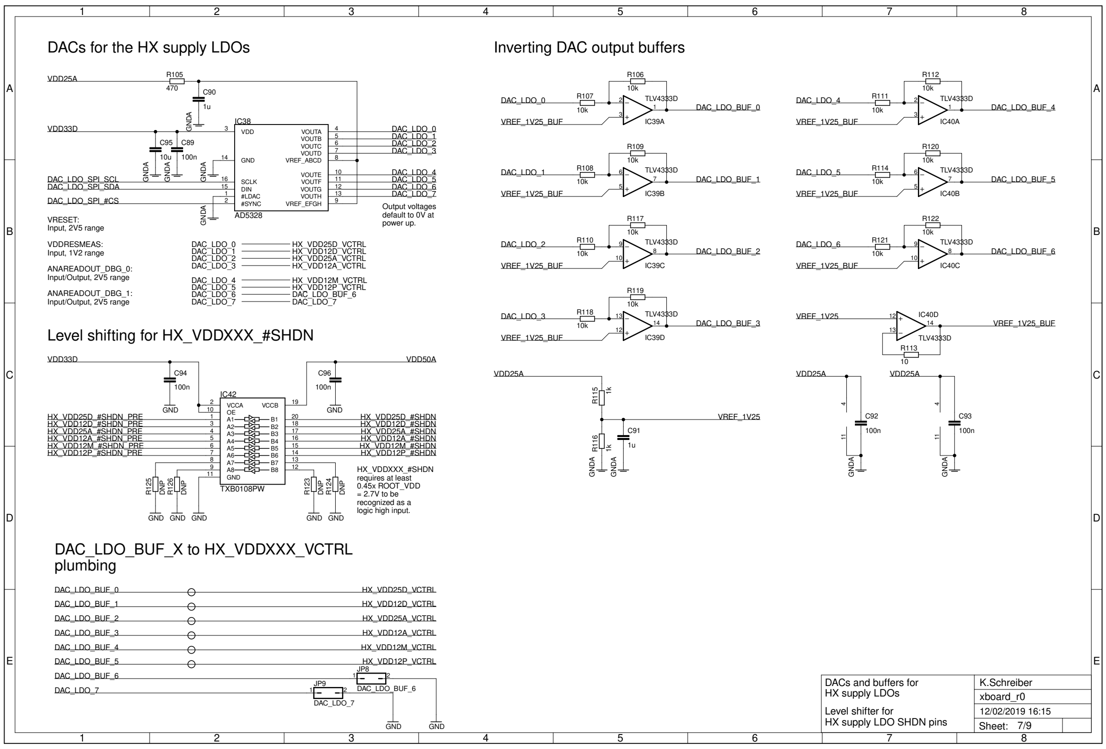
SPI Interface / GPIO level shifters
FPGA_CLK_RX_X_FROM_HICANN can be used as digital I/O with 1V8 logic level.
SPI Interface / GPIO level shifters
FPGA_CLK_TX_* come from differential line drivers.
SPI pin extension for mux configuration and state LEDs
These LED pins are intended to alternatively serve as GPOs.
JTAG interface and reset
JTAG and RESET_L logic levels can be set to either 1V8 or 3V3 with a jumper on the iBoard adapter board. To avoid another level shifter on the xBoard, IC36 can be left open. In that case, one of the following two actions MUST be taken instead:
- Connect 2.5 V to the middle postion of jumper JP2 (HICANN VDD Jumper) on the iBoard adapter board
- Add a voltage divider instead of a jumper at JP2 (HICANN VDD Jumper) on the iBoard adapter board:
VDD33 --[39R]-- HICANN_VDD --[33R]-- VDD18
I2C distribution
Occupied I2C addresses:
0110011 10000XX
1010XXX 100010X
I2C level from iBoard Adapter Board is 3V3.
Pull-up resistors are 22k.
Connection to iBoard
Pin name spacing
SPI level shifters
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 3/9

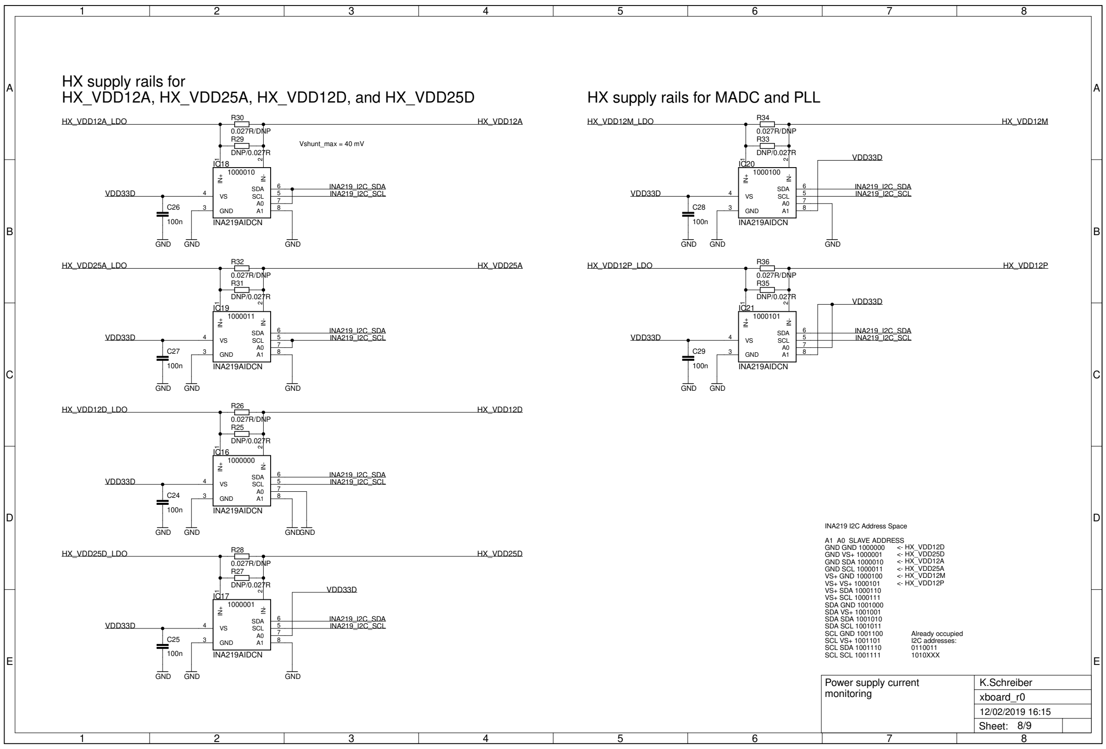


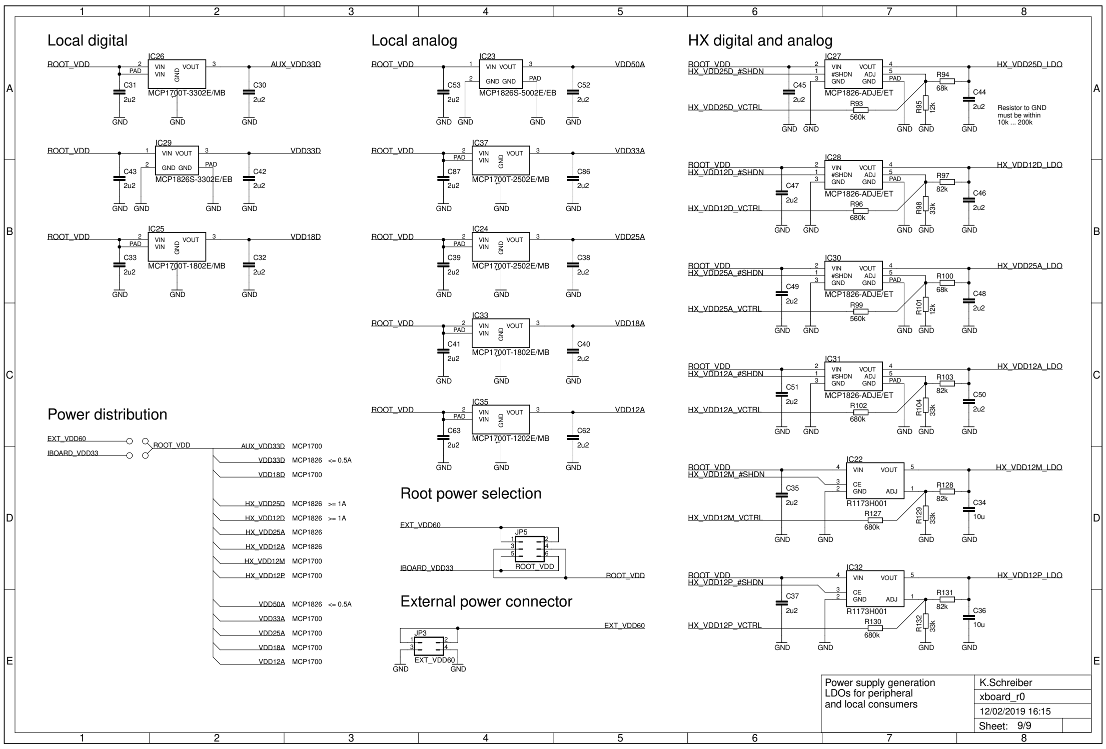
DAC for 1V2 and 2V5 signal ranges.
VRESET: Input, 2V5 range
VDDRESMEAS: Input, 1V2 range
ANAREADOUT_DBG_0: Input/Output, 2V5 range
ANAREADOUT_DBG_1: Input/Output, 2V5 range
Output voltages default to 0V at power up.
DAC output buffers and V-I-converters
VGS_max ~ 0.7 V
VGS_typ ~ 0.5 V
Analog input switches and IREF measurement resistor
DACs and buffers
Analog switches
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 4/9


Analog readout mux and amplifier
AUX preamp output
ADC preamp output
Differential readout and debug
TS5A3357
LMH6612
LMH6518
0 .. 2.5 V
2.391 .. 2.717 V
VCM_IN = 1.9 .. 3.1 V
ADC_VCM goes to FLYSPI as well.
Input levels are below 2.5 V
Readout amplifier and analog muxes
SPI GPO extension
Analog readout connectors
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 5/9


ADC322X - fast ADC for FlySpi readout chain replacement
PLL reference clock conversion
ADC322X can be configured to stream out data on either one or two differential lanes per channel. In the default assembly configuration (R58/59/60/61), both lanes of Channel A are connected to the FPGA ports while Channel B remains disconnected.
ADC3221 25 MSPS
ADC3222 50 MSPS
ADC3223 80 MSPS
ADC3224 125 MSPS
TXB0108PW
MCP1700T-3002E/MB
SN65LVDTXX has internal 110R termination.
SN65LVDT34D
CLK level shifter and ADC for analog readout
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 6/9

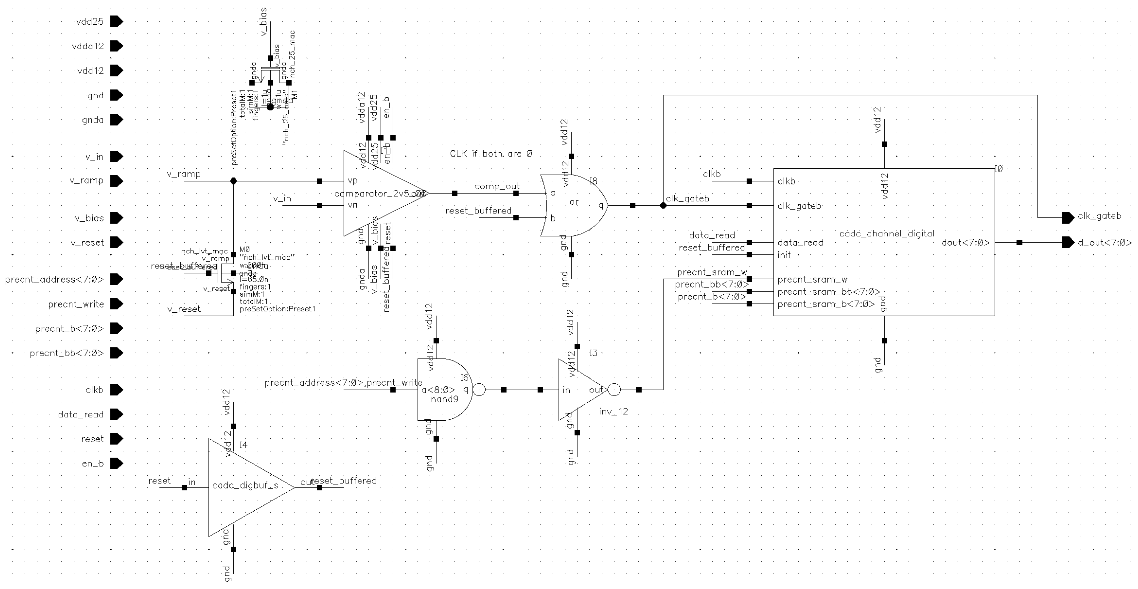

DACs for the HX supply LDOs
Inverting DAC output buffers
Level shifting for HX_VDDXXX_#SHDN
DAC_LDO_BUF_X to HX_VDDXXX_VCTRL plumbing
AD5328
TXB0108PW
TLV4333D
Output voltages default to 0V at power up.
VRESET: Input, 2V5 range
VDDRESMEAS: Input, 1V2 range
ANAREADOUT_DBG_0: Input/Output, 2V5 range
ANAREADOUT_DBG_1: Input/Output, 2V5 range
HX_VDDXXX_#SHDN requires at least 0.45x ROOT_VDD = 2.7V to be recognized as a logic high input.
DACs and buffers for HX supply LDOs
Level shifter for HX supply LDO SHDN pins
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 7/9


HX supply rails for HX_VDD12A, HX_VDD25A, HX_VDD12D, and HX_VDD25D
HX supply rails for MADC and PLL
Vshunt_max = 40 mV
INA219AIDCN
INA219 I2C Address Space
A1 A0 SLAVE ADDRESS
GND GND 1000000 <- HX_VDD12D
GND VS+ 1000001 <- HX_VDD25D
GND SDA 1000010 <- HX_VDD12A
GND SCL 1000011 <- HX_VDD25A
VS+ GND 1000100 <- HX_VDD12M
VS+ VS+ 1000101 <- HX_VDD12P
VS+ SDA 1000110
VS+ SCL 1000111
SDA GND 1001000
SDA VS+ 1001001
SDA SDA 1001010
SDA SCL 1001011
SCL GND 1001100
SCL VS+ 1001101
SCL SDA 1001110
SCL SCL 1001111
Already occupied I2C addresses: 0110011 1010XXX
Power supply current monitoring
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 8/9

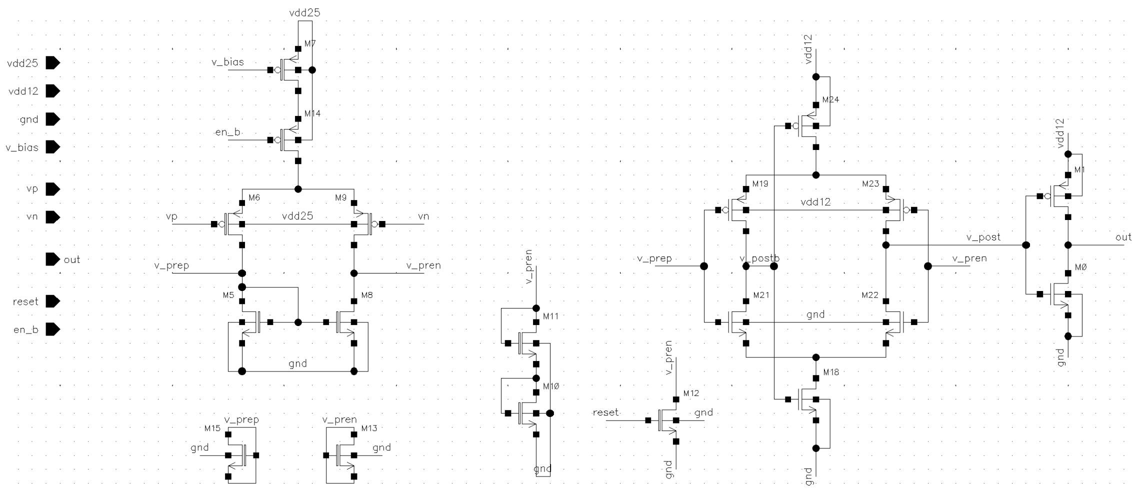

Local digital
Local analog
HX digital and analog
Power distribution
Root power selection
External power connector
IC26 MCP1700T-3302E/MB
IC29 MCP1826S-3302E/EB
IC25 MCP1700T-1802E/MB
IC23 MCP1826S-5002E/EB
IC37 MCP1700T-2502E/MB
IC24 MCP1700T-2502E/MB
IC33 MCP1700T-1802E/MB
IC35 MCP1700T-1202E/MB
IC27 MCP1826-ADJE/ET
IC28 MCP1826-ADJE/ET
IC30 MCP1826-ADJE/ET
IC31 MCP1826-ADJE/ET
IC22 R1173H001
IC32 R1173H001
ROOT_VDD
AUX_VDD33D
VDD33D
VDD18D
VDD50A
VDD33A
VDD25A
VDD18A
VDD12A
HX_VDD25D_LDO
HX_VDD12D_LDO
HX_VDD25A_LDO
HX_VDD12A_LDO
HX_VDD12M_LDO
HX_VDD12P_LDO
Resistor to GND must be within 10k ... 200k
EXT_VDD60
IBOARD_VDD33
AUX_VDD33D MCP1700
VDD33D MCP1826 <= 0.5A
VDD18D MCP1700
HX_VDD25D MCP1826 >= 1A
HX_VDD12D MCP1826 >= 1A
HX_VDD25A MCP1826
HX_VDD12A MCP1826
HX_VDD12M MCP1700
HX_VDD12P MCP1700
VDD50A MCP1826 <= 0.5A
VDD33A MCP1700
VDD25A MCP1700
VDD18A MCP1700
VDD12A MCP1700
JP5
JP3
Power supply generation LDOs for peripheral and local consumers
K.Schreiber
xboard_r0
12/02/2019 16:15
Sheet: 9/9

C.3 CADC

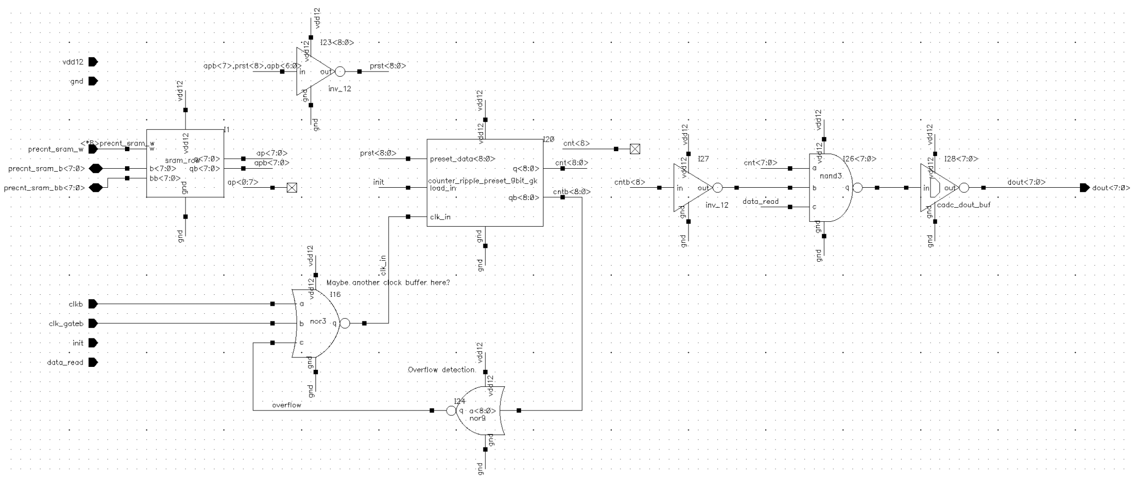
C.3.1 Channel



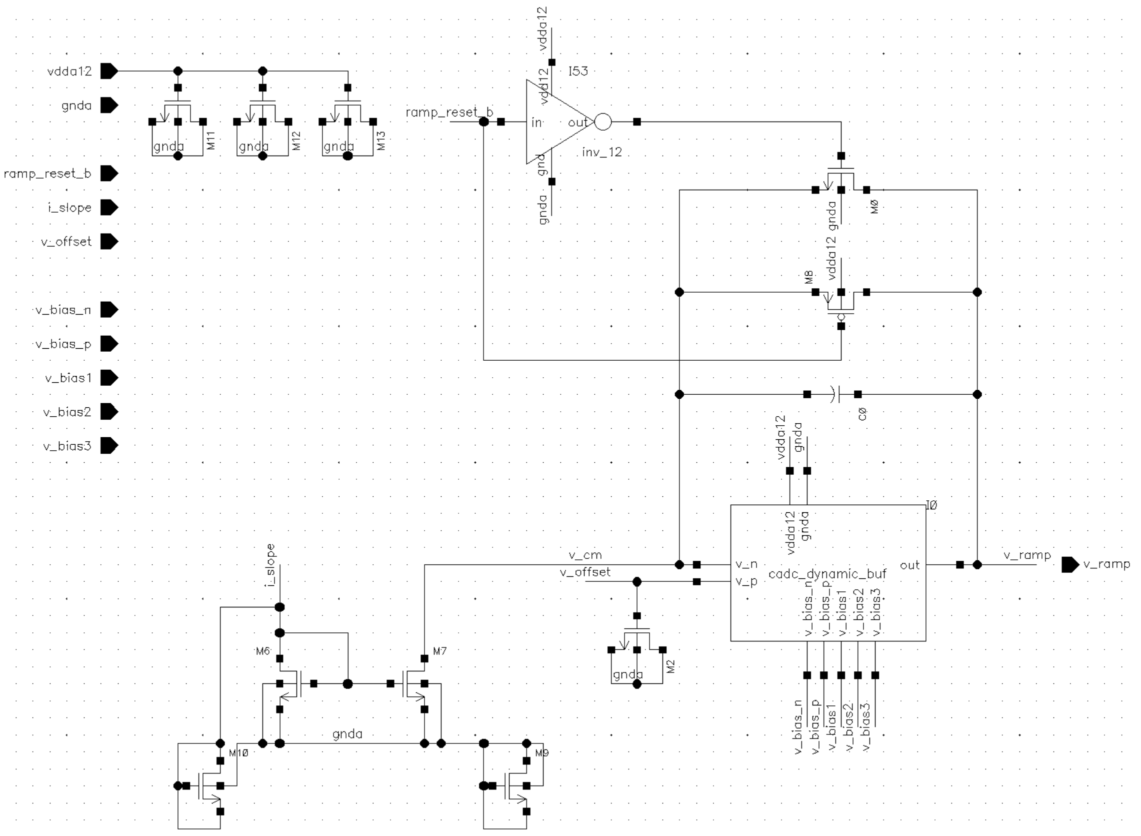
C.3.2 Comparator



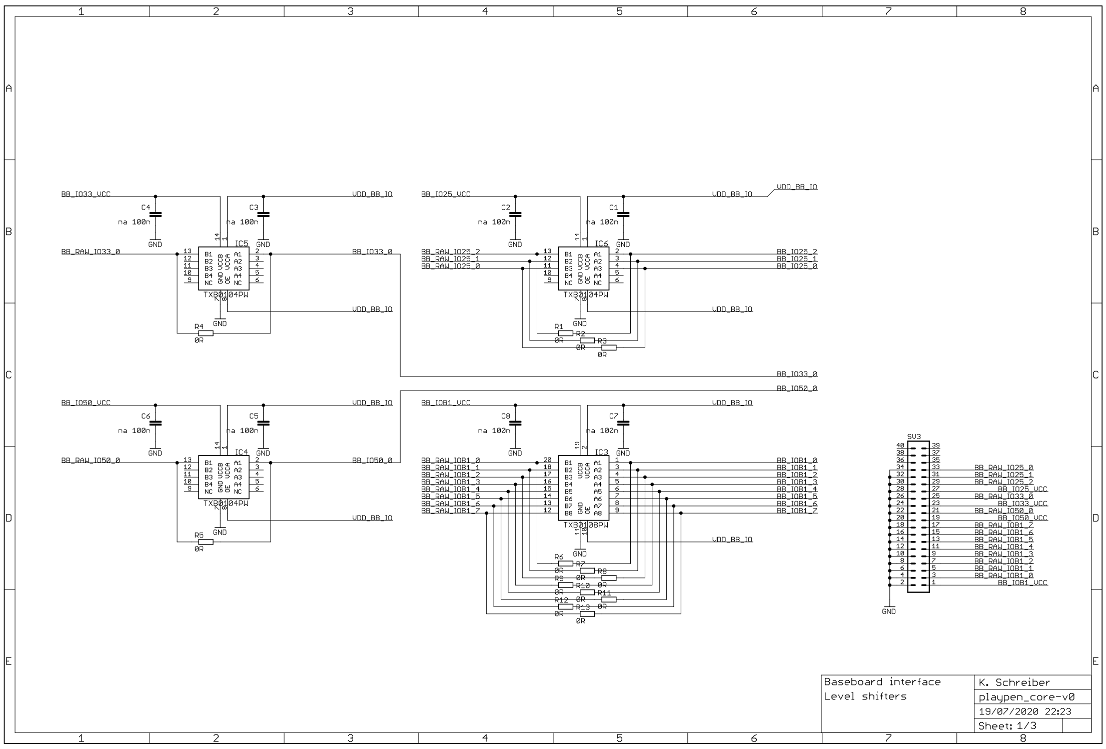
C.3.3 Channel (digital)

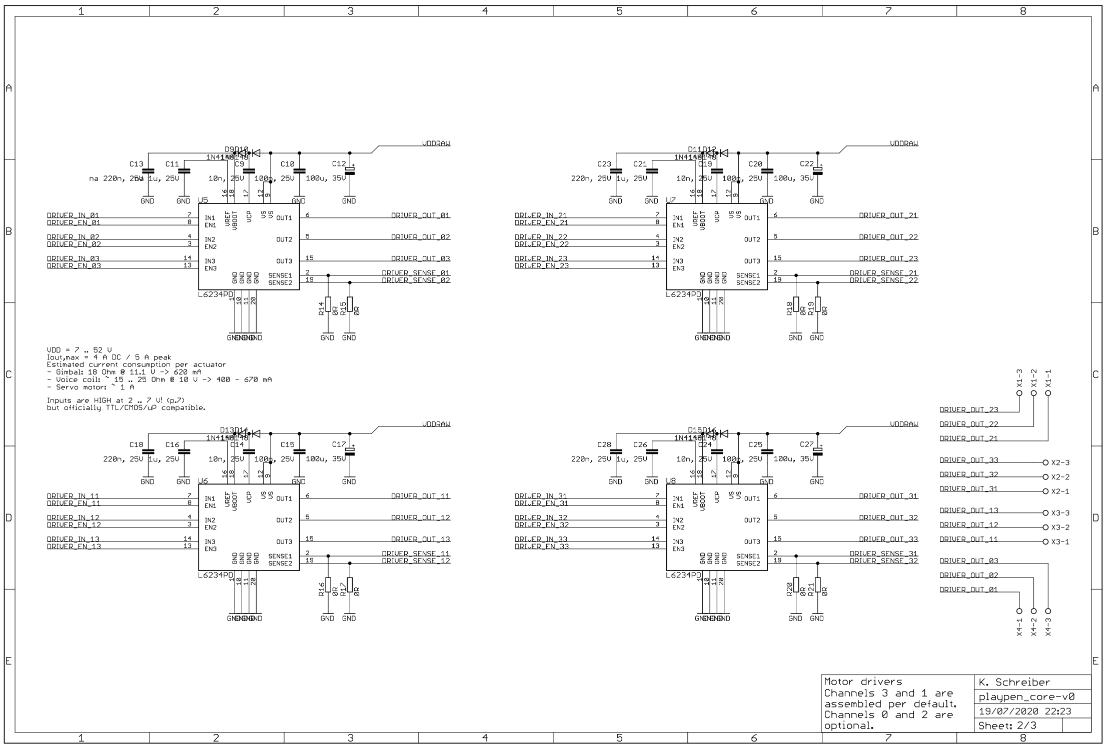


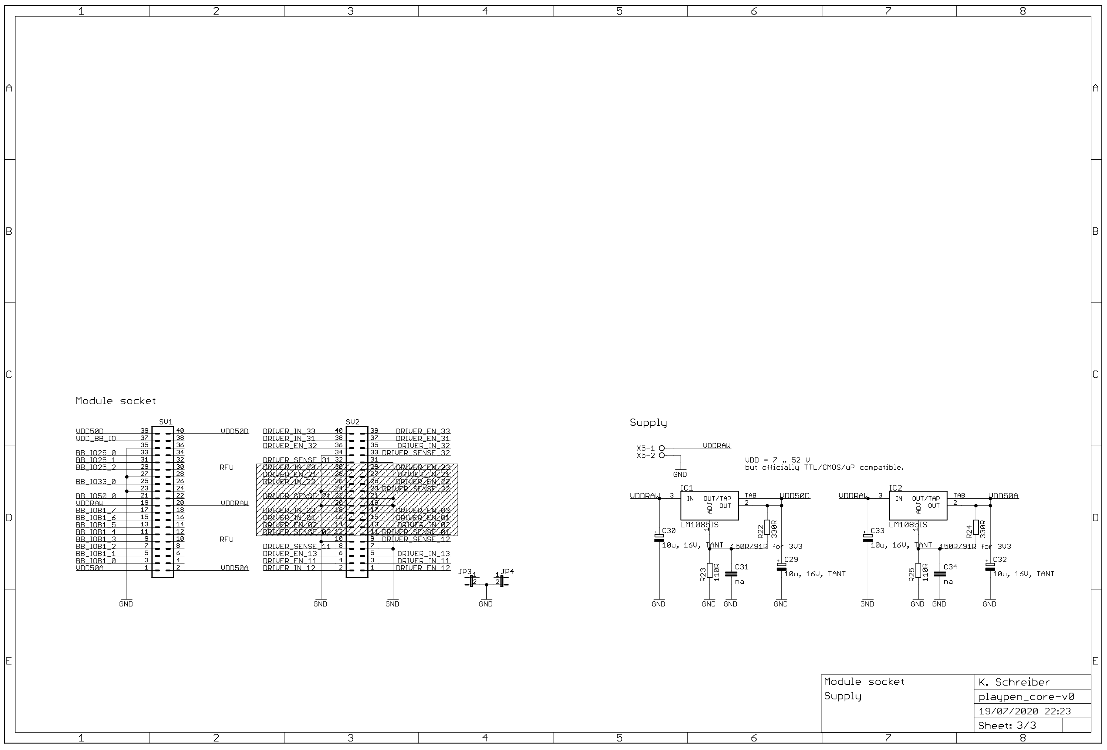
C.3.4 Ramp generator



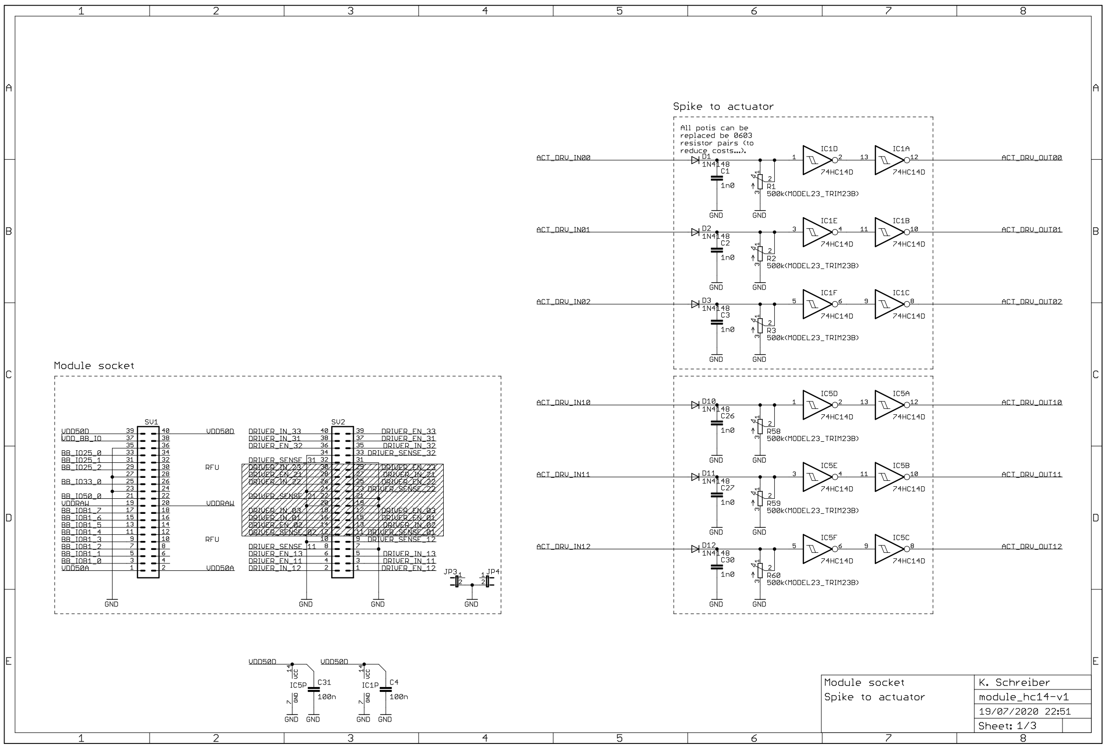
C.4 PlayPen-1: core module

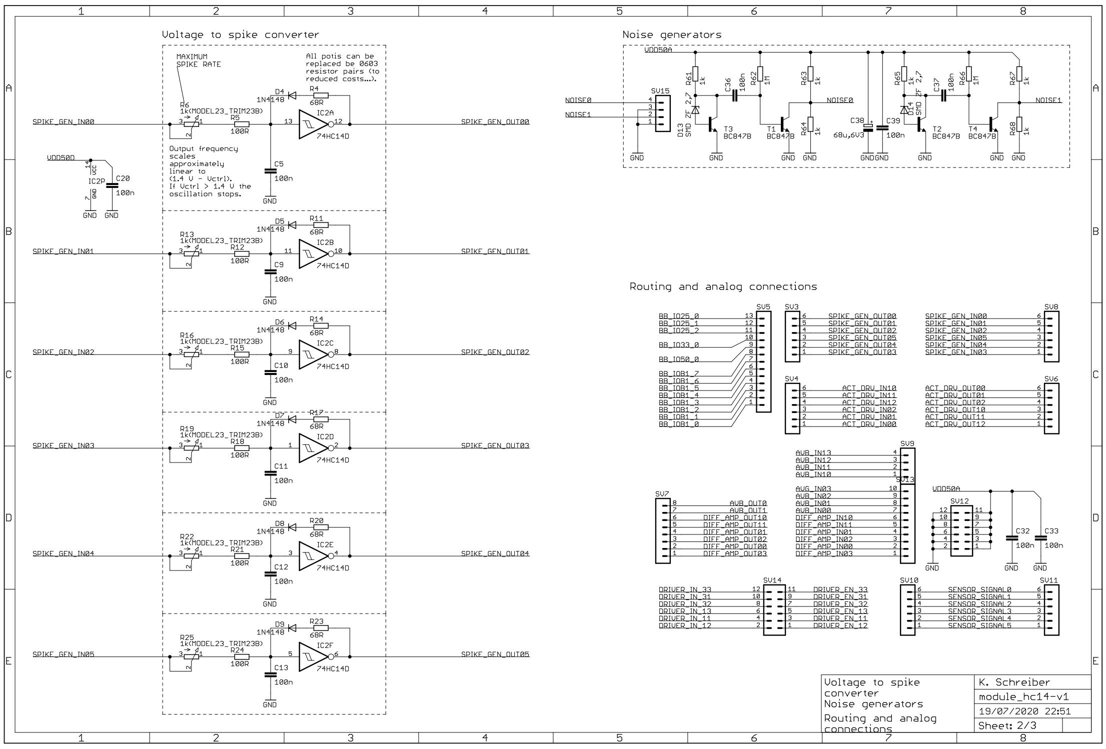


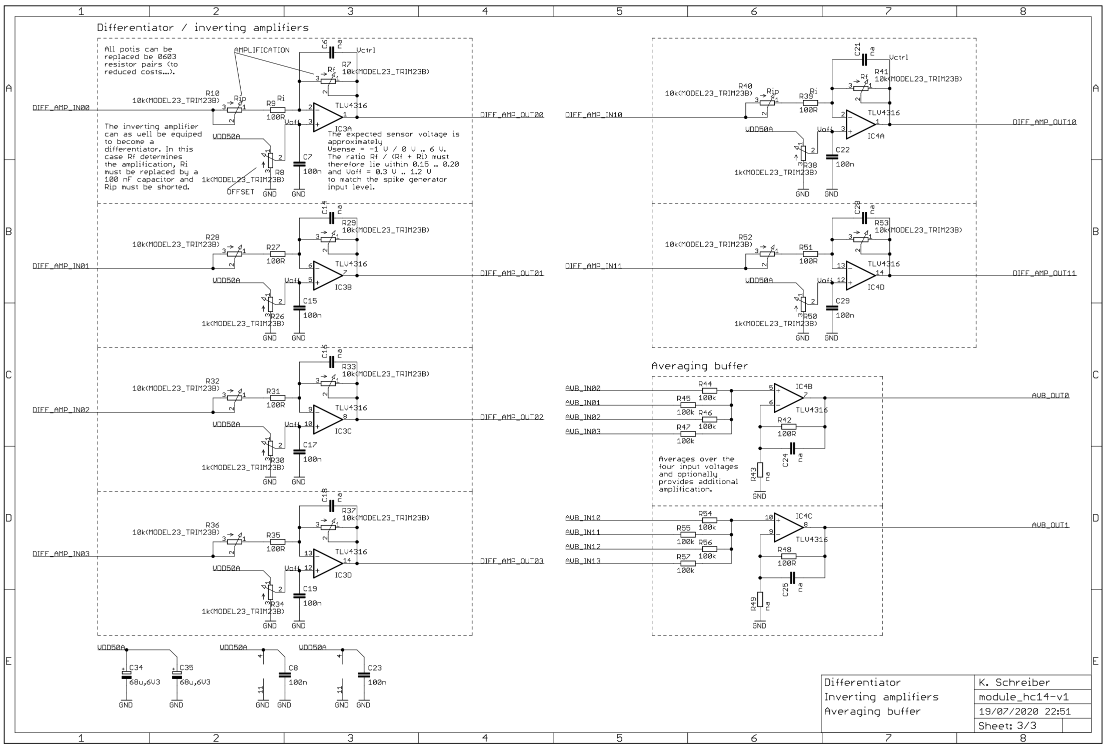

VDD = 7 .. 52 V
Iout,max = 4 A DC / 5 A peak
Estimated current consumption per actuator
- Gimbal: 10 Ohm @ 11.1 V -> 620 mA
- Voice coil: ~ 15 .. 25 Ohm @ 10 V -> 400 - 670 mA
- Servo motor: ~ 1 A
Inputs are HIGH at 2 .. 7 V! (p.7)
but officially TTL/CMOS/uP compatible.
Motor drivers
Channels 3 and 1 are assembled per default.
Channels 0 and 2 are optional.
K. Schreiber
playpen_core-v0
19/07/2020 22:23
Sheet: 2/3


Module socket
SV1
VDD50D
VDD_BB_IO
BB_IO25_0
BB_IO25_1
BB_IO25_2
BB_IO33_0
BB_IO50_0
VDDRAW
BB_IOB1_7
BB_IOB1_6
BB_IOB1_5
BB_IOB1_4
BB_IOB1_3
BB_IOB1_2
BB_IOB1_1
BB_IOB1_0
VDD50A
RFU
GND
SV2
DRIVER_IN_33
DRIVER_IN_31
DRIVER_EN_32
DRIVER_SENSE_31
DRIVER_IN_23
DRIVER_EN_21
DRIVER_IN_22
DRIVER_SENSE_21
DRIVER_IN_03
DRIVER_IN_01
DRIVER_EN_02
DRIVER_SENSE_02
DRIVER_SENSE_11
DRIVER_EN_13
DRIVER_EN_11
DRIVER_IN_12
DRIVER_EN_33
DRIVER_EN_31
DRIVER_IN_32
DRIVER_SENSE_32
DRIVER_EN_23
DRIVER_IN_21
DRIVER_EN_22
DRIVER_SENSE_22
DRIVER_EN_03
DRIVER_EN_01
DRIVER_IN_02
DRIVER_SENSE_01
DRIVER_SENSE_12
DRIVER_IN_13
DRIVER_IN_11
DRIVER_EN_12
JP3
JP4
Supply
X5-1
X5-2
VDDRAW
VDD = 7 .. 52 V
but officially TTL/CMOS/uP compatible.
IC1
IN
OUT/TAP
OUT
ADJ
TAB
LM1085IS
VDD50D
C30
10u, 16V, TANT
R22
330R
150R/91R for 3V3
R23
110R
C31
na
C29
10u, 16V, TANT
IC2
VDD50A
C33
R24
R25
C34
C32
Module socket
Supply
K. Schreiber
playpen_core-v0
19/07/2020 22:23
Sheet: 3/3

C.5 PlayPen-1: signal processing module




Voltage to spike converter
Noise generators
Routing and analog connections
Voltage to spike converter
Noise generators
Routing and analog connections
K. Schreiber
module_hc14-v1
19/07/2020 22:51
Sheet: 2/3



Differentiator / inverting amplifiers

All potis can be replaced be 0603 resistor pairs (to reduced costs...).

The inverting amplifier can as well be equiped to become a differentiator. In this case Rf determines the amplification, Ri must be replaced by a 100 nF capacitor and Rip must be shorted.

The expected sensor voltage is approximately Vsense = -1 V / 0 V .. 6 V. The ratio Rf / (Rf + Ri) must therefore lie within 0.15 .. 0.20 and Voff = 0.3 V .. 1.2 V to match the spike generator input level.

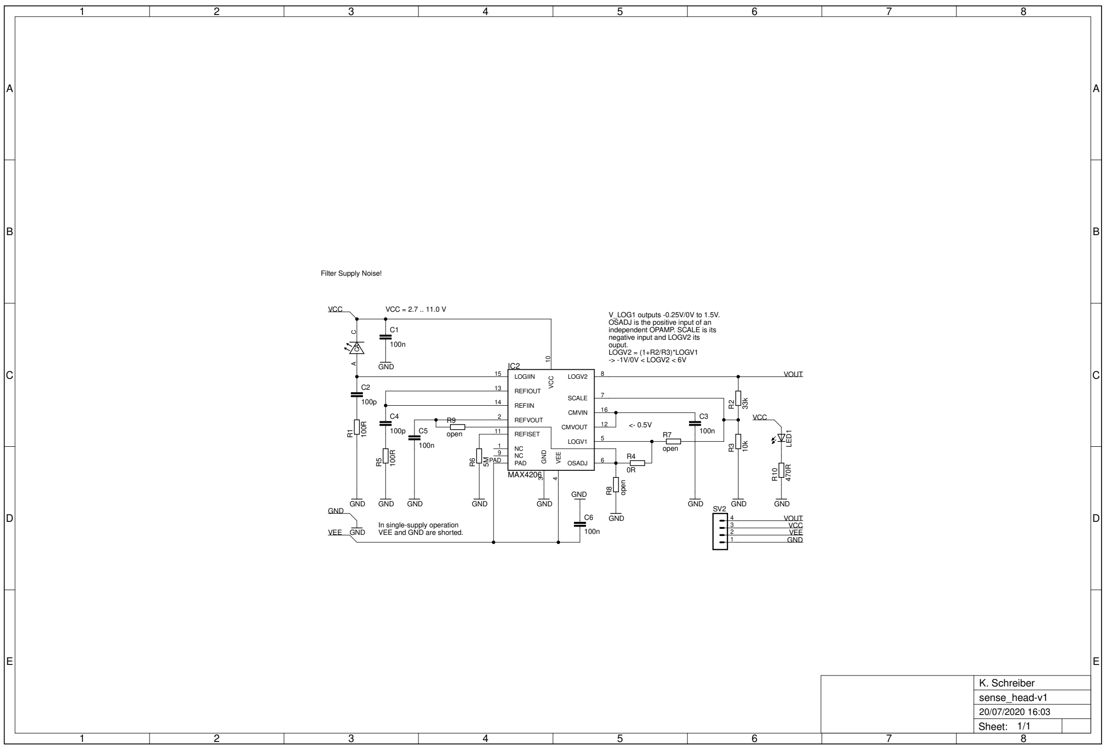
Averaging buffer

Averages over the four input voltages and optionally provides additional amplification.

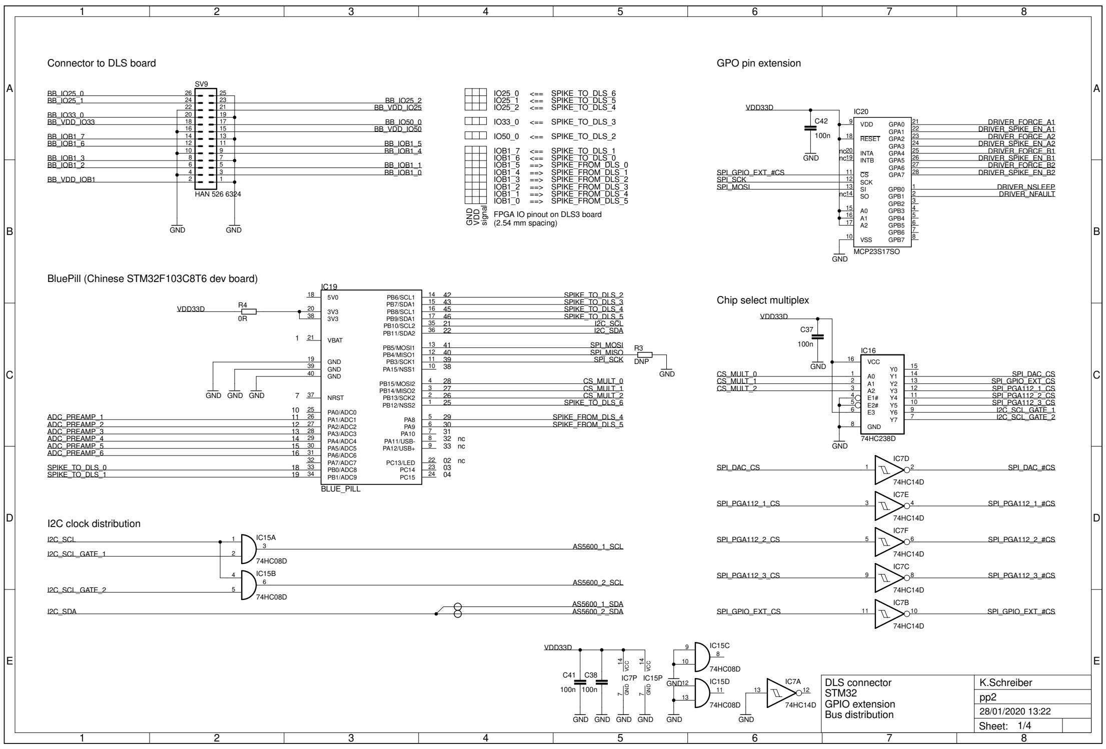
Differentiator
Inverting amplifiers
Averaging buffer

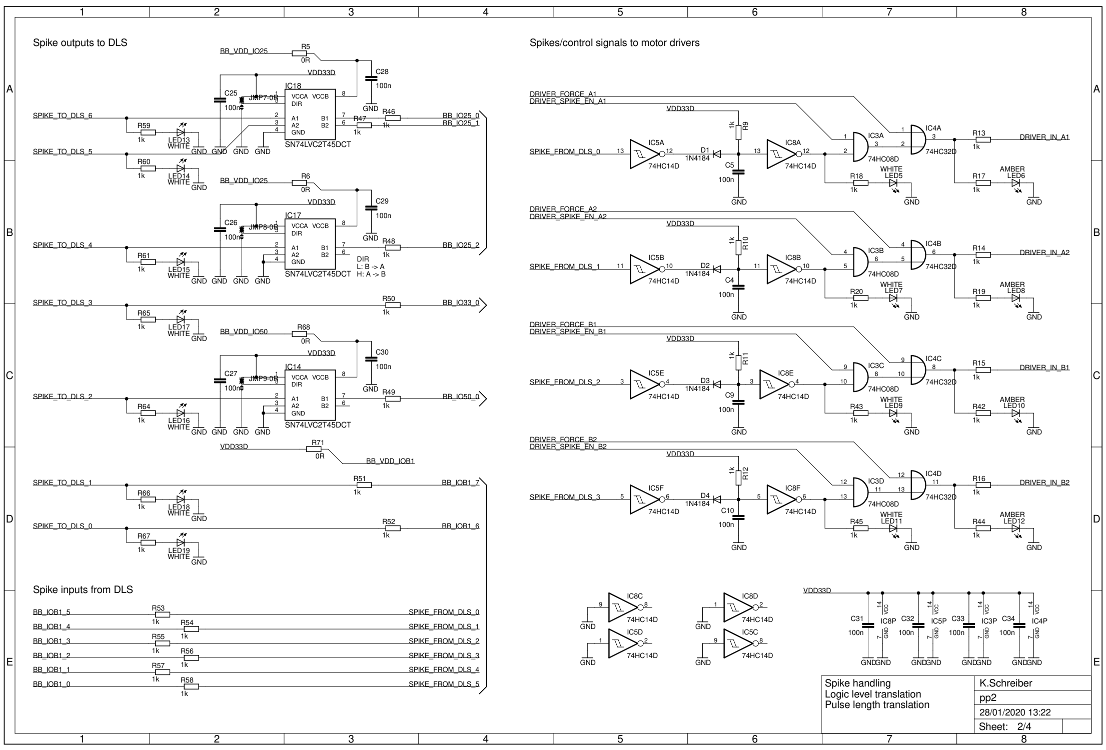
K. Schreiber
module_hc14-v1
19/07/2020 22:51
Sheet: 3/3

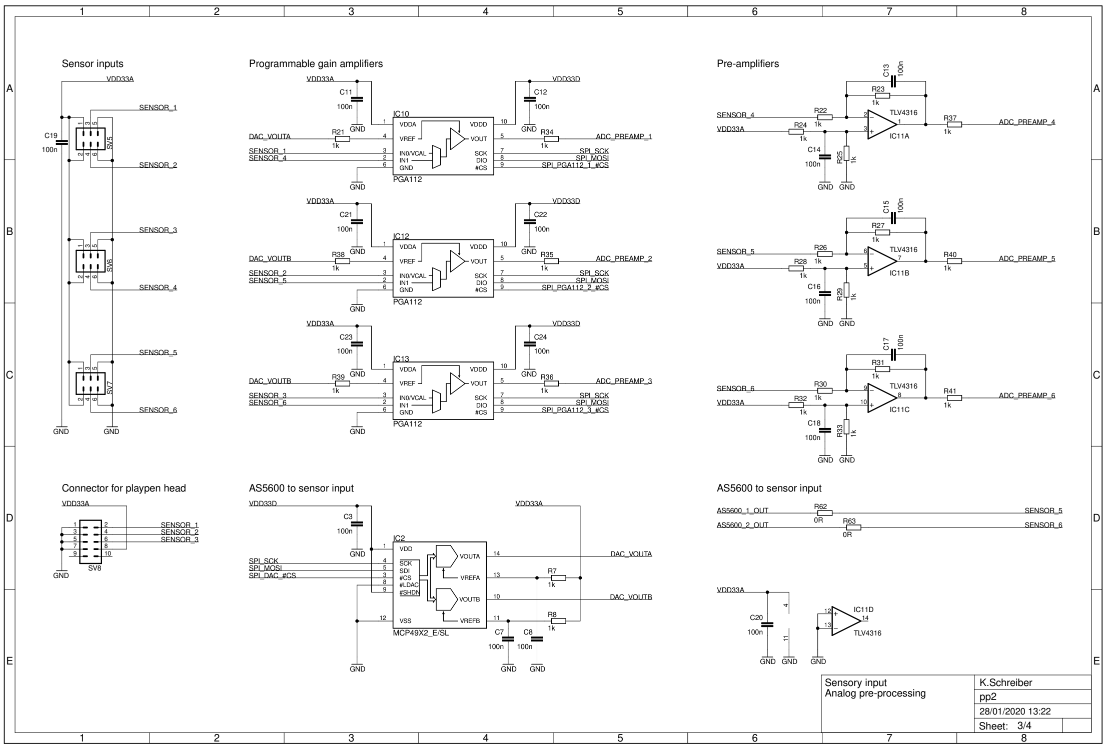
C.6 PlayPen-1: sensor head

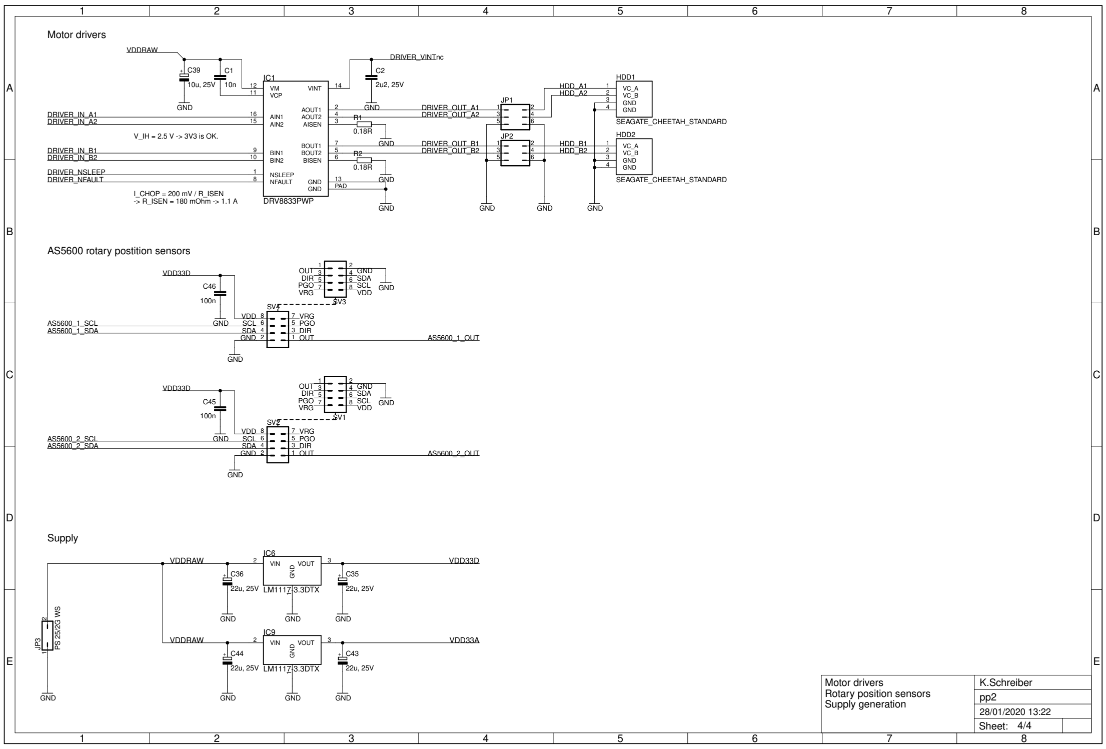


C.7 PlayPen-2

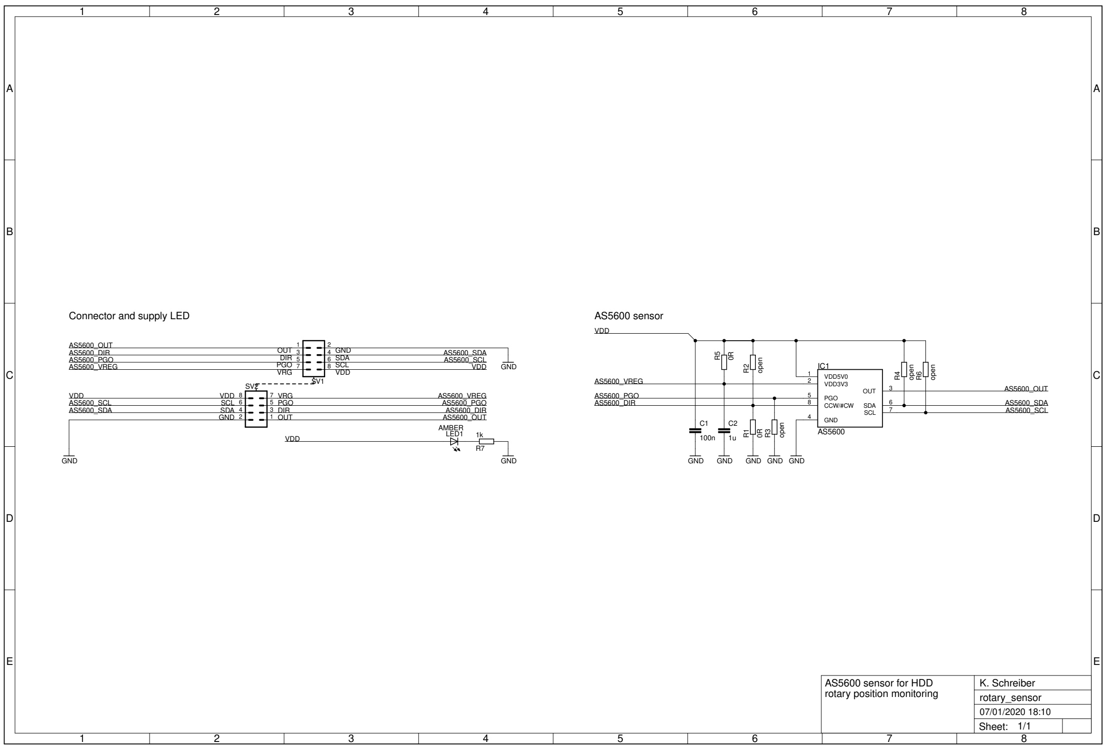

Connector to DLS board
GPO pin extension
BluePill (Chinese STM32F103C8T6 dev board)
Chip select multiplex
I2C clock distribution
FPGA IO pinout on DLS3 board
(2.54 mm spacing)
DLS connector
STM32
GPIO extension
Bus distribution
K.Schreiber
pp2
28/01/2020 13:22
Sheet: 1/4


Spike outputs to DLS
Spikes/control signals to motor drivers
Spike inputs from DLS
Spike handling
Logic level translation
Pulse length translation
K.Schreiber
pp2
28/01/2020 13:22
Sheet: 2/4

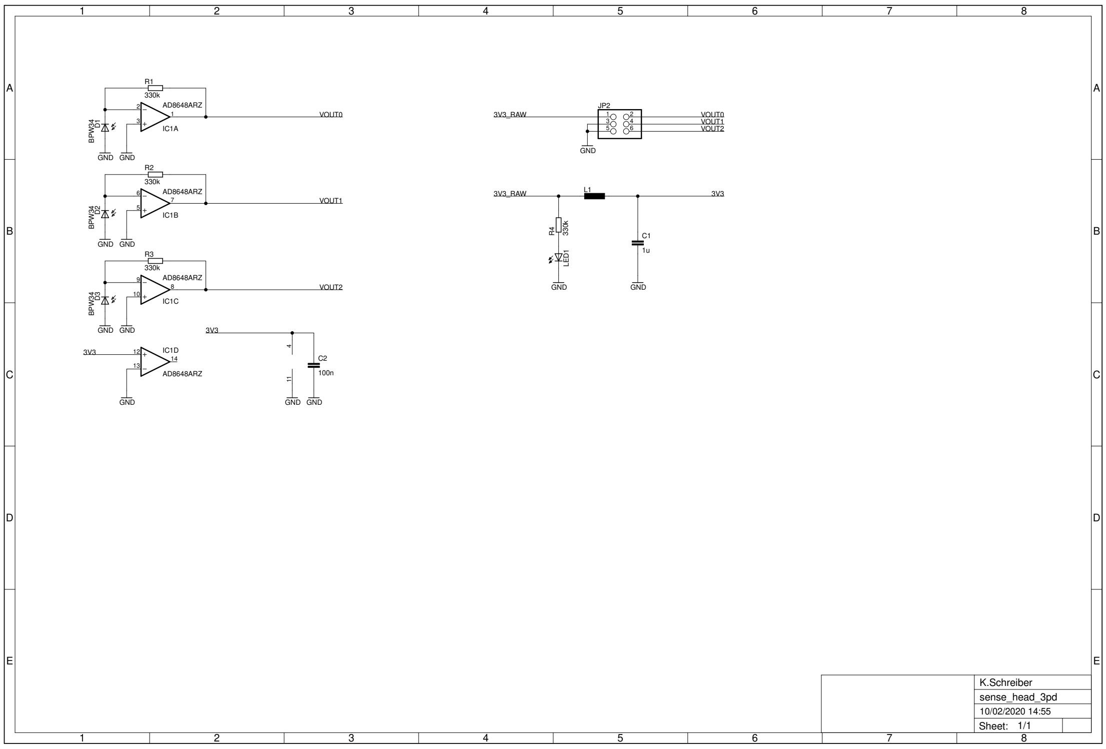

Sensor inputs
Programmable gain amplifiers
Pre-amplifiers
Connector for playpen head
AS5600 to sensor input
AS5600 to sensor input
VDD33A
VDD33D
SENSOR_1
SENSOR_2
SENSOR_3
SENSOR_4
SENSOR_5
SENSOR_6
SV5
SV6
SV7
SV8
C19
100n
IC10
IC12
IC13
PGA112
DAC_VOUTA
DAC_VOUTB
SPI_SCK
SPI_MOSI
SPI_PGA112_1_#CS
SPI_PGA112_2_#CS
SPI_PGA112_3_#CS
ADC_PREAMP_1
ADC_PREAMP_2
ADC_PREAMP_3
ADC_PREAMP_4
ADC_PREAMP_5
ADC_PREAMP_6
TLV4316
IC11A
IC11B
IC11C
IC11D
IC2
MCP49X2_E/SL
SPI_DAC_#CS
AS5600_1_OUT
AS5600_2_OUT
R62
R63
0R
GND
Sensory input
Analog pre-processing
K.Schreiber
pp2
28/01/2020 13:22
Sheet: 3/4


Motor drivers
AS5600 rotary postition sensors
Supply
VDDRAW
DRIVER_VINTnc
IC1
DRV8833PWP
C39
10u, 25V
C1
10n
C2
2u2, 25V
DRIVER_IN_A1
DRIVER_IN_A2
DRIVER_IN_B1
DRIVER_IN_B2
DRIVER_NSLEEP
DRIVER_NFAULT
V_IH = 2.5 V -> 3V3 is OK.
I_CHOP = 200 mV / R_ISEN
-> R_ISEN = 180 mOhm -> 1.1 A
R1
R2
0.18R
DRIVER_OUT_A1
DRIVER_OUT_A2
DRIVER_OUT_B1
DRIVER_OUT_B2
JP1
JP2
HDD_A1
HDD_A2
HDD_B1
HDD_B2
HDD1
HDD2
SEAGATE_CHEETAH_STANDARD
VDD33D
C46
C45
100n
AS5600_1_SCL
AS5600_1_SDA
AS5600_2_SCL
AS5600_2_SDA
AS5600_1_OUT
AS5600_2_OUT
SV1
SV2
SV3
SV4
IC6
IC9
LM1117-3.3DTX
C35
C36
C43
C44
22u, 25V
VDD33D
VDD33A
JP3
PS 25/2G WS
GND
Motor drivers
Rotary position sensors
Supply generation
K.Schreiber
pp2
28/01/2020 13:22
Sheet: 4/4

C.8 PlayPen-2: rotary sensor



C.9 PlayPen-2: sensor head



Acknowledgements

I would like to thank

Prof. Karlheinz Meier for allowing me to conduct my doctoral studies in his outstanding Electronic Vision(s) group, for the selfless conscientiousness which he dedicated to every single aspect of his work, and for enthusiastically supporting my off-track endeavor to implement neuromorphic insect brains. I wish he could have witnessed the numerous magnificent experimental results that his BrainScaleS-2 project produced during the last two years.

Dr. Johannes Schemmel for being my supervisor, for providing me with the freedom that was necessary to pursue the topics in chapters 4 and 5, for generously supporting me in difficult situations, and for taking over Prof. Meier's challenging job of skillfully navigating the Electronic Vision(s) group.

Prof. Michael Hausmann for reviewing this work and Prof. Kurt Roth and Prof. Ulrich Schwarz for participating in my oral examination.

Sebastian and Yannik for proof-reading.

Philipp for invaluable compiler support relating to the insect project and Christian for implementing the Spike I/O relating to the PlayPen project.

My closer colleagues Benjamin, Christian, Gerd, Paul, Sebastian, Timo, and Yannik for many productive, entertaining, interesting, and warmhearted hours in our offices, Café Botanik, Sardinia, or Austria, etc..

All other dear and unique fellow visionaries for being kind colleagues and brilliant professionals: Ákos, Andreas B., Andreas G., Andreas H., Aron, Björn, Christian M., Christoph, Dan, David, Eric, Hartmut, Johann, Johannes W., Joscha, José, Julian, Frau Kleveta, Laura, Lars, Markus, Maurice, Mihai, Mitja, Oliver, Philipp D., Philipp S., Ralf, Sebastian S., Simeon, Tobias, Vitali, etc..

In particular, once more, Sebastian for his friendship, boundless helpfulness, diligence, and remarkable excellence in literally everything he engages in.

My mom for always wisely guiding and supporting me.

Ruyi for her many encouragements, her love and care, and for her ability to outshine even the darkest moments with cheerfully smiling cordiality.

Statement of originality (Erklärung):

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 selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 27.10.2020 ....