Simulator-Like Exploration of Cortical Network Architectures with a Mixed-Signal VLSI System

Daniel Brüderle*, Johannes Bill*, Bernhard Kaplan*, Jens Kremkow†,
Karlheinz Meier*, Eric Müller* and Johannes Schemmel*
*Kirchhoff Institute for Physics, Ruperto-Carola University, Heidelberg, Germany
†Neurobiology and Biophysics, Albert-Ludwig University, Freiburg, Germany
Email: bruederle@kip.uni-heidelberg.de

***Abstract*— In this paper we describe our approach towards highly configurable neuromorphic hardware systems that serve as useful and flexible tools in modeling neuroscience. We utilize a mixed-signal VLSI model that implements a massively accelerated network of spiking neurons, and we describe a novel methodological framework that allows to exploit both the speed and the programmability of this device for the systematic and simulator-like exploration of cortical network architectures. We present a variety of experimental results that illustrate the functionality of our modeling platform, and we verify all hardware measurements with reference software simulations. Especially on the network level these comparison studies are unique in terms of the quantitative correspondence between the data. The presented hardware experiments include high-conductance states in hardware neurons and the application of synaptic depression and facilitation for self-adjusting network architectures.**

I. Introduction

Neuromorphic hardware devices mimic the structure and emulate the function of biological neural networks in a physical form. With a tradition going back to the 1980s [1], today an active neuromorphic engineering community is developing analog or mixed-signal VLSI models of neural systems [2]–[8]. Compared to numerical simulations with digital computers, the main advantage of this physical approach arises from the locally analog and massively parallel nature of the computations. This leads to the fact that neuromorphic network models are typically highly scalable, i.e. they can emulate neural networks in real time or much faster, independent of the realized network size. A clear disadvantage of such systems is the limited flexibility of the implemented models. Typically, neuron and synapse parameters as well as the network connectivity can be programmed only to a certain degree and always within limited ranges. Furthermore, inter-chip event-communication bandwidths set practical limits on the scaling of network sizes [8], [9].

In contrast to most neuromorphic engineering projects, the hardware systems we consider in this article exhibit a speedup factor of up to 10^5 compared to the emulated biological real time [7], [8]. This massive acceleration and an implementation path towards large-scale network architectures with low power consumption [8] make such neuromorphic systems potentially valuable research tools for the modeling neuroscience community. There, software simulators are commonplace [10], but often impose performance limitations [11].

In order to establish neuromorphic hardware devices as useful components within the toolboxes of neural network modelers, a proof of such systems' biological relevance plus an interface that provides the operability by non-hardware-experts are required. The novel methodological framework we presented in [12], [13] satisfies these two conditions to a large extent: It allows to utilize a neuromorphic system for the flexible, simulator-like exploration of network architectures and dynamics. In this paper we shortly describe the experimental platform, i.e. the hardware device and the applied methods and software concepts for its operation. Then we present a selection of experimental results that indicate the practical options emerging from the symbiotic operation of the highly programmable device and the software structure it is embedded into.

II. Materials and Methods

A. The Neuromorphic System

The employed mixed-signal VLSI device [7], [14] implements leaky integrate-and-fire neuron models with conductance-based synapses, designed to exhibit a linear correspondence with existing conductance-based modeling approaches [15]. The chip was built on a single 25 mm^2 die using a standard 180 nm CMOS process. It models networks of up to 384 neurons and the temporal evolution of the weights of 10^5 configurable synapses. The system can be operated with an acceleration factor of up to 10^5 while recording the neural action potentials with a temporal resolution of approximately $0.3\,\mathrm{ns}$, which corresponds to $30\,\mu\mathrm{s}$ in the interpreted biological time domain. For a precise description of the neuron and synapse circuits, the connectivity model and the implemented long-term and short-term synaptic plasticity features see [7], [13] and [14].

B. The Operating Paradigm

1) Integration into PyNN: PyNN is a simulator-independent, Python-based language that allows to describe neural network models [16]. It offers functions and classes for the setup and control of experiments, and it provides standard cell models and standardized units. PyNN supports various software simulators like NEURON [17], NEST [18], PCSIM [19] and more. With PyNN, which is free, open source and well documented, a user can set up a neural network model,

978-1-4244-5309-2/10/$26.00 ©2010 IEEE

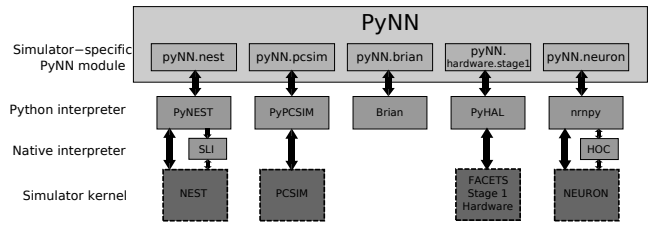


Fig. 1. Schematic of the simulator-independent modeling language PyNN.

run it on any of the supported back-ends without changing the code, and directly compare the results. This provides the possibility to conveniently port experiments between different simulators, to transparently share models and results, and to verify data acquired from different back-ends by direct comparison (see Figure 1). As one important step towards a novel neuromorphic modeling platform, the operating software framework for the hardware system described above has been integrated into the PyNN concept [12], [13].

2) Simulator-Based Calibration: The hardware-specific PyNN approach incorporates quantitative translation methods between the neuromorphic system dynamics and its biological interpretation, both in terms of electrical variables and different time domains. It also incorporates calibration routines that minimize the impact of transistor-level fluctuations onto the behavior of neural and synaptic circuits, directly using reference software simulations for the biologically relevant gauging of hardware parameters. These calibration methods, their PyNN-based implementation and the results achieved with these efforts are described in detail in [13].

C. Experimental Verification with Simulator Reference

In order to prove the functionality of our approach, we want to provide data that can be transparently judged. Therefore, we exploit the benefits offered by the integration of the utilized hardware system into the PyNN paradigm. In direct and quantitative comparison with the software simulators NEST and PCSIM we perform experiments with the neuromorphic device. One PyNN description per experiment is sufficient, only a single line of code has to be changed to choose the utilized back-end. The results acquired with the different platforms can then be checked against each other without further modifications, as will be shown in the following section.

III. Experiments

With the neuromorphic modeling platform presented in Section II, we performed a variety of experiments. Here we provide a selection of these experiments in the form of brief setup outlines, result samples and references to more elaborate descriptions. All hardware data are given in units that reflect their biological interpretation, i.e. the hardware-specific PyNN module automatically translated the original hardware measurement values with linear transformations [13]. This is indicated by the labels `BVD` and `BTD`, which stand for biological voltage and time domain, respectively.

A. Single Cell Experiments

All analog circuits are subject to electronic phenomena like noise, crosstalk or transistor-level variations and therefore should never be expected to deliver a perfect matching with numerical simulations in terms of membrane voltages or spike times. Nevertheless, directly comparing membrane potentials, spike times and firing rates of individual neurons can provide deepened insights into hardware-specific model characteristics.

1) Membrane Potentials and Spikes: Both on the hardware system and in NEST, a single neuron was stimulated with $N_\mathrm{e} = 48$ Poisson-type spike trains via excitatory plus $N_\mathrm{i} = 16$ spike trains via inhibitory synapses. All stimulation spike trains fired with a frequency $f_\mathrm{in} = 8\,\mathrm{Hz}$ (`BTD`) for the full experiment duration of $T_\mathrm{exp} = 5\,\mathrm{s}$ (`BTD`), resulting in an output rate of the stimulated cell of approximately 3 Hz (`BTD`), which is realistic for cortical cells in awake mammals. The membrane potential and the output spikes of the stimulated neuron were recorded. A hardware run that matches the NEST result well in terms of a spike train difference measure described in [13] was selected for the membrane potential comparison shown in Figure 2. See [13] for a detailed description of the experiment, including all applied neuron and synapse parameter values. A more systematic hardware vs. NEST spike-time comparison can also be found in [12], [20].

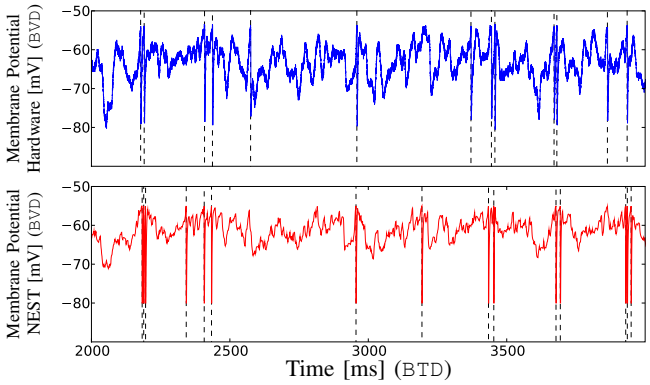
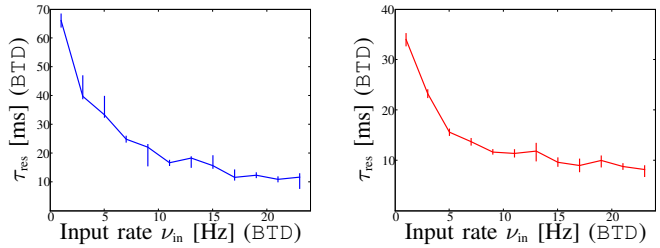


Fig. 2. The sub-threshold membrane potential of a neuron stimulated by Poisson-type spike trains as emulated with the hardware system (top) and as simulated with NEST (bottom). Both in NEST and in the hardware model, action potentials are not modeled with depolarization peaks, but only registered in terms of their occurrence time and followed by a reset mechanism. Therefore, the output spike times of the recorded neuron are indicated by vertical dashed lines.

2) High-Conductance States in Hardware Neurons: In [21] we introduced a purely spike-based method that allows to determine the ability of a neuron's membrane to act as a coincidence detector. We showed that, based on this method, we can experimentally determine the critical level of synaptic stimulation that is necessary to establish a so-called high-conductance state in our hardware neurons [22]. In this state the low-pass filtering effect of the membrane is minimized, and we generated it by stimulating a single cell with a set of Poisson-type input spike trains via excitatory and inhibitory synapses. In Figure 3 the temporal resolution τ_res of a neuron,

defined in and measured according to [21], is plotted as a function of the firing rate ν_{in} of the applied input spike trains - both for a hardware and a NEST neuron. The number and



(a) Hardware (b) NEST

Fig. 3. Temporal resolution τ_{res} of a neuron membrane as a function of its synaptically induced conductance, which is varied via the external stimulation rate ν_{in} [21].

ratio of excitatory and inhibitory spike trains were dynamically adjusted such that only the input firing rate determines the total synaptically induced membrane conductance, while the average output firing rate of the stimulated neuron was kept within a narrow regime.

B. Exploring Network Architectures

The presented neuromorphic modeling platform allows to explore neural network architectures systematically and with sufficient statistics - both being typically weak points of numerical simulation approaches.

1) Recurrent Cortical Architectures: We studied the firing dynamics in a recurrent, cortically inspired network architecture [23]. For a complete and full description of the conducted experiment series, including all precise parameter values, see [13]. In Figure 4a the setup is described: A population of excitatory and a population of inhibitory neurons were randomly with fixed probabilities and stimulated with excitatory Poisson-type spike trains. All synapses were static, i.e. neither facilitating nor depressing. The synaptic weight of all excitatory connections was kept constant, while the weight of all inhibitory synapses was varied. The stimulation firing rate was also swept, and the thereby defined two-dimensional parameter space was systematically explored. Figures 4b and 4c show the resulting average network firing rates for every considered point in this parameter space, as emulated with the hardware system and computed with NEST.

In [13], further statistical descriptors for comparing the network dynamics generated with the two different platforms are considered, namely measures for the synchrony and the regularity of firing within the network. We show that very different activity regimes can be generated with the hardware, and that for large parameter regions the hardware firing dynamics are in accordance with NEST.

2) Self-Adjusting Networks: We also implemented another, similar network setup, again consisting of an excitatory and an inhibitory population, both randomly connected internally and among each other. This time the stimulation was both excitatory and inhibitory, and we enabled short-term plasticity (facilitation/depression) in all recurrent synapses, according to a principle suggested in [24] and illustrated in Figure 5a. The hardware-compatible implementation of this specific connection pattern is described in detail in [25]. With the dynamic synapses enabled, these networks remain in a biologically relevant and technically trouble-free activity regime for a large variety of different input strengths and variabilities –see Figure 5c and 5d– while their capability to process information is not lost (not shown here). If the dynamic synapses are replaced by static ones, the network activity strongly amplifies itself and consequently gets critically high for most of the studied input parameter regions, see Figure 5b. Hence, the use of the dynamic synapses in the described way provides a self-balancing network architecture, which can be useful to maintain biologically relevant regimes and counterbalance hardware-specific variations on the network level.

IV. Conclusion

The data we presented illustrates the functionality of our approach towards flexible, scalable and massively accelerated neuroscientific modeling tools based on mixed-signal neuromorphic devices. Integrating hardware interfaces into the PyNN concept builds a bridge between the communities of neuromorphic engineers and computational neuroscientists, who have been working in rather disjoint projects so far. We believe that also other groups who develop hardware models of spiking neural networks could benefit from the proposed paradigm. Considering the presented framework and data, we are optimistic that in near future neuromorphic systems can not only form artificial neural systems for a broad field of technological applications, but actually contribute new insights into neural information processing in the brain.

Acknowledgment

This work is supported by the European Union under the grant no. IST-2005-15879 (FACETS).

References

[1] C. A. Mead, *Analog VLSI and Neural Systems*. Reading, MA: Addison Wesley, 1989.

[2] G. Indiveri, E. Chicca, and R. Douglas, "A VLSI array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity," *IEEE Transactions on Neural Networks*, vol. 17, no. 1, pp. 211–221, Jan 2006.

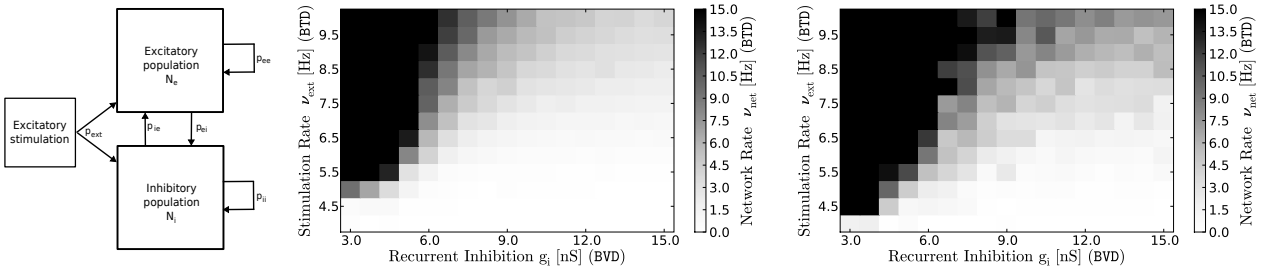
[3] P. A. Merolla and K. Boahen, "Dynamic computation in a recurrent network of heterogeneous silicon neurons," in *Proceedings of the IEEE ISCAS*, 2006.

[4] R. J. Vogelstein, U. Mallik, J. T. Vogelstein, and G. Cauwenberghs, "Dynamically reconfigurable silicon array of spiking neuron with conductance-based synapses," *IEEE Transactions on Neural Networks*, vol. 18, pp. 253–265, 2007.

[5] P. Häfliger, "Adaptive WTA with an analog VLSI neuromorphic learning chip," *IEEE Transactions on Neural Networks*, vol. 18, no. 2, pp. 551–72, 2007.

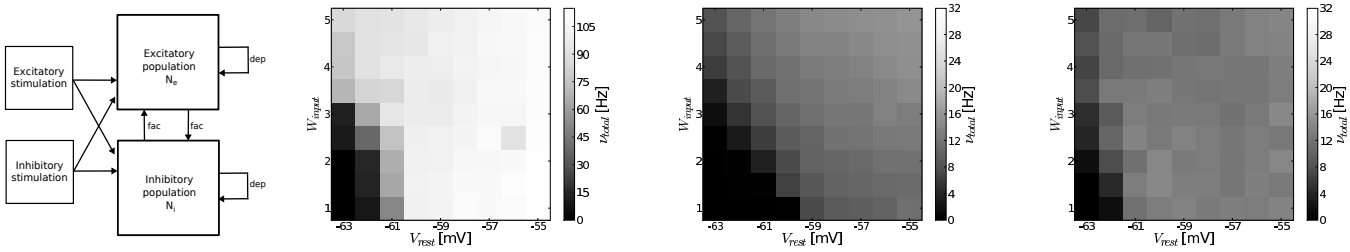
[6] S. Renaud, J. Tomas, Y. Bornat, A. Daouzli, and S. Saghi, "Neuromimetic ICs with analog cores: an alternative for simulating spiking neural networks," in *Proceedings of the IEEE ISCAS*, 2007.

[7] J. Schemmel, D. Brüderle, K. Meier, and B. Ostendorf, "Modeling synaptic plasticity within networks of highly accelerated I&F neurons," in *Proceedings of the IEEE ISCAS*, 2007.



(a) Setup (b) Hardware: Firing Rate ν_{net} (c) NEST: Firing Rate ν_{net}

Fig. 4. In (a) a schematic diagram of the recurrent network architecture is shown. The labels p_{xy} for each arrow indicate the probability of making a synapse of a neuron in population x onto a neuron in population y, where e stands for *excitatory* ($N_{\text{e}} = 120$) and i for *inhibitory* ($N_{\text{i}} = 40$). The externally generated Poisson processes are used to activate the neurons, p_{ext} is the probability for every possible stimulus-to-neuron connection to be established. In (b) the average network firing rate ν_{net} is plotted as a function of the externally applied stimulus frequency ν_{ext} and the strength g_{i} of the inhibitory feedback synapses, implemented with the hardware system. In (c) the corresponding results as computed with NEST are plotted. Every data tile represents the average over 200 experiments of 10 s (BTD) duration each. For each of these 200 runs the network structure and the stimulation spike patterns were randomly re-generated according to the parameter values given.



(a) Setup (b) Hardware: Static synapses (c) Hardware: Dynamic synapses (d) PCSIM: Dynamic synapses

Fig. 5. (a) shows a schematic diagram of the recurrent network architecture. Both within the excitatory ($N_{\text{e}} = 144$) and the inhibitory ($N_{\text{i}} = 48$) neuron pool, the intra-population synapses are depressing, while all inter-population connections are facilitating. The strength and variability of external stimulation is controlled via the neuron resting potential parameter V_{rest} and a scaling factor W_{input} applied to the weights of the static stimulation synapses. For every data tile, 20 randomly connected networks with new external stimulation were generated. The resulting average firing rates are illustrated by different shades of gray in (b) to (d). In (b), which shows hardware measurements, all recurrent synapses are static resulting in high firing rates, of which the maximum values are mostly determined by an upper bandwidth limitation of the hardware spike recording circuitry. In (c) and (d) the recurrent synapses are facilitating or depressing according to (a), which results in a stable, biologically relevant activity regime for a major part of the considered parameter space.

[8] J. Schemmel, J. Fieres, and K. Meier, "Wafer-scale integration of analog neural networks," in *Proceedings of the IJCNN*, 2008.

[9] H. K. O. Berge and P. Häfliger, "High-speed serial AER on FPGA," in *Proceedings of the 2007 IEEE ISCAS*, 2007, pp. 857–860.

[10] R. Brette et al., "Simulation of networks of spiking neurons: A review of tools and strategies," *Journal of Computational Neuroscience*, vol. 3, no. 23, pp. 349–98, December 2006.

[11] A. Morrison, C. Mehring, T. Geisel, A. Aertsen, and M. Diesmann, "Advancing the boundaries of high connectivity network simulation with distributed computing," *Neural Comput.*, vol. 17, no. 8, pp. 1776–1801, 2005.

[12] D. Brüderle, E. Müller, A. Davison, E. Muller, J. Schemmel, and K. Meier, "Establishing a novel modeling tool: A python-based interface for a neuromorphic hardware system," *Front. Neuroinform.*, vol. 3, no. 17, 2009.

[13] D. Brüderle, "Neuroscientific modeling with a mixed-signal VLSI hardware system," Ph.D. dissertation, 2009. [Online]. Available: http://www.ub.uni-heidelberg.de/archiv/9656

[14] J. Schemmel, A. Grübl, K. Meier, and E. Muller, "Implementing synaptic plasticity in a VLSI spiking neural network model," in *Proceedings of the IJCNN*, 2006.

[15] A. Destexhe, "Conductance-based integrate-and-fire models," *Neural Comput.*, vol. 9, no. 3, pp. 503–514, 1997.

[16] A. P. Davison, D. Brüderle, J. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger, "PyNN: a common interface for neuronal network simulators," *Front. Neuroinform.*, vol. 2, no. 11, 2008.

[17] M. L. Hines and N. T. Carnevale, *The NEURON Book*. Cambridge, UK: Cambridge University Press, 2006.

[18] M.-O. Gewaltig and M. Diesmann, "NEST (NEural Simulation Tool)," *Scholarpedia*, vol. 2, no. 4, p. 1430, 2007.

[19] D. A. Pecevski, T. Natschläger, and K. N. Schuch, "PCSIM: A parallel simulation environment for neural circuits fully integrated with Python," *Front. Neuroinform.*, pending publication.

[20] D. Brüderle, A. Grübl, K. Meier, E. Muller, and J. Schemmel, "A software framework for tuning the dynamics of neuromorphic silicon towards biology," in *Proceedings of the IWANN*, 2007, pp. 479–486.

[21] B. Kaplan, D. Brüderle, J. Schemmel, and K. Meier, "High-conductance states on a neuromorphic hardware system," in *Proceedings of the IJCNN*, 2009.

[22] A. Destexhe, M. Rudolph, and D. Pare, "The high-conductance state of neocortical neurons in vivo," *Nature Reviews Neuroscience*, vol. 4, pp. 739–751, 2003.

[23] N. Brunel, "Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons," *Journal of Computational Neuroscience*, vol. 8, no. 3, pp. 183–208, 2000.

[24] D. Sussillo, T. Toyoizumi, and W. Maass, "Self-tuning of neural circuits through short-term synaptic plasticity," *J Neurophysiol*, vol. 97, no. 6, pp. 4079–4095, 2007.

[25] J. Bill, "Self-stabilizing network architectures on a neuromorphic hardware system," Diploma thesis, 2008. [Online]. Available: http://www.kip.uni-heidelberg.de/Veroeffentlichungen/details.php?id=1893