frontiers in Neuroscience

ORIGINAL RESEARCH
published: 24 February 2022
doi: 10.3389/fnins.2022.795876

The BrainScaleS-2 Accelerated Neuromorphic System With Hybrid Plasticity

*Christian Pehle†, Sebastian Billaudelle†, Benjamin Cramer†, Jakob Kaiser†, Korbinian Schreiber†, Yannik Stradmann†, Johannes Weis†, Aron Leibfried, Eric Müller and Johannes Schemmel**

Electronic Visions, Kirchhoff-Institute for Physics, Heidelberg University, Heidelberg, Germany

Since the beginning of information processing by electronic components, the nervous system has served as a metaphor for the organization of computational primitives. Brain-inspired computing today encompasses a class of approaches ranging from using novel nano-devices for computation to research into large-scale neuromorphic architectures, such as TrueNorth, SpiNNaker, BrainScaleS, Tianjic, and Loihi. While implementation details differ, spiking neural networks—sometimes referred to as the third generation of neural networks—are the common abstraction used to model computation with such systems. Here we describe the second generation of the BrainScaleS neuromorphic architecture, emphasizing applications enabled by this architecture. It combines a custom analog accelerator core supporting the accelerated physical emulation of bio-inspired spiking neural network primitives with a tightly coupled digital processor and a digital event-routing network.

Keywords: neuromorphic computing, physical modeling, neuroscientific modeling, spiking neural network accelerator, plasticity

OPEN ACCESS

Edited by:
Qilin Hua,
Beijing Institute of Nanoenergy and Nanosystems (CAS), China

Reviewed by:
Arash Ahmadi,
Carleton University, Canada
Jason Eshraghian,
University of Michigan, United States

***Correspondence:**
Johannes Schemmel
schemmel@kip.uni-heidelberg.de

†These authors have contributed equally to this work

Specialty section:
This article was submitted to Neuromorphic Engineering, a section of the journal Frontiers in Neuroscience

Received: *15 October 2021*
Accepted: *27 January 2022*
Published: *24 February 2022*

Citation:
Pehle C, Billaudelle S, Cramer B, Kaiser J, Schreiber K, Stradmann Y, Weis J, Leibfried A, Müller E and Schemmel J (2022) The BrainScaleS-2 Accelerated Neuromorphic System With Hybrid Plasticity. Front. Neurosci. 16:795876. doi: 10.3389/fnins.2022.795876

1. INTRODUCTION

One important scientific goal of computational neuroscience is the advancement of brain-inspired computing. Continuous-time emulators for modeling brain function play an essential role in this endeavor. They provide resource-efficient platforms for the bottom-up modeling of brain function—including computationally expensive aspects like plasticity and learning or structured neurons. BrainScaleS is a neuromorphic computing platform that realizes this approach to the furthest extent possible with current technologies by constructing a physical replica of the most commonly used reductionist view of the biological brain: a network of neurons connected via plastic synapses.

In this aspect, it differs from most other modeling approaches within the computational neuroscience community. In particular, while the network model operates, no differential equation gets solved. Biological processes are not represented by discrete-time changes of a multitude of bits representing some binary approximation of molecular biology. Instead, the temporal evolution of physical quantities, such as current and voltage, directly correspond to the neural dynamics in BrainScaleS. In that regard, our approach is similar to system architectures using novel nano-devices to perform computation. However, we focus on creating a controllable, configurable substrate based on well-understood CMOS technology, which can serve as a platform for research into system-level aspects of such an approach.

In designing the BrainScaleS-2 architecture, we had several high-level design goals and use cases in mind. Some of them are informed by the limitations we discovered with previous system designs. The overarching design goal was to enable large-scale accelerated emulation of spiking neural networks. This requires a *scalable* system architecture. The system we will describe in this article is a *unit of scale* for such a large-scale system. Analog neuromorphic hardware presents unique challenges to scalability compared to digital neuromorphic hardware. Since the constituting components are subject to both fixed-pattern noise, as well as temperature and time-dependent drift of parameters, architectural solutions are needed to address these. Device variations (fixed-pattern noise) are addressable by calibration, but this requires acceptable parameter ranges and sufficient compute resources, which enable constant scaling with system size. A novel design for analog parameter storage (Hock et al., 2013) ensures wide parameter ranges and stability of parameters over time. Rapid calibration is enabled by including embedded processor cores and high-resolution and multi-channel analog-to-digital converters with access to all relevant analog states. Taken together, they enable *on-chip calibration* in our unit of scale and ensure that calibration time remains constant with system size.

From a user perspective of the system, we aim to support several operation modes and use cases for the system architecture. They can be distinguished along several dimensions. Perhaps the most common mode of operation is to perform experiments in *batch mode*. In this mode, experiment instances are queued and sequentially executed on the system without any data dependency among the different instances. Batch-mode execution is used for parameter sweeps, evaluation of biology-inspired learning rules, or inference once task-specific parameters for a classification task have been found. There are several ways to introduce data dependencies between experiment instances. We refer to them collectively as *in-the-loop* operation. Our system enables us to close these loops on several time scales and at different points of the system hierarchy, as we will further describe in Section 2.3. This ability features also prominently in some of the experiments: among them the learning-to-learn approach (Bohnstingl et al., 2019), briefly discussed in Section 3.2, the surrogate gradient in-the-loop optimization described in Section 3.3.1, and analog artificial neural network training in Section 3.3.2.

One of the key distinguishing features of the architecture is our approach to synaptic plasticity. In contrast to designs like Loihi (Davies et al., 2018), which only supports micro-coded operations per synapse or other designs with fixed plasticity, we support plasticity programs with complex control and data dependencies. The combination of massive-parallel data acquisition of (analog) system observables (synaptic correlation and membrane voltage traces) and the efficient, digital evaluation in programmable plasticity rules is a unique strength of our system. This approach will be described in more detail in Section 2.3 and is the basis of all experiments reported in Section 3.2.

Another key distinguishing feature is that our synaptic crossbar can process weighted spikes. This capability enables the use of the same components to implement analog vector-matrix multiplication. We will give an overview of the analog emulation of artificial neural networks on our system in Section 2.2.

The rest of this paper is structured as follows: We begin with a more detailed description of the system architecture in Section 2.1. Afterwards, we describe the analog core of the system in Section 2.2, in particular the design of the neuron circuitry with an emphasis on the design decisions that lead to controllability and a wide range of biological parameters. Beyond the analog core, the system also incorporates two loosely coupled embedded processors enabling the realization of *hybrid plasticity* schemes, emulation of virtual environments for reinforcement experiments, as well as the orchestration of calibration and data transfer. We describe this part of the system in Section 2.3.

Taken together, these design decisions and features of the system architecture enable the use of the system on a wide range of tasks and operation modes. The current system design can serve as a versatile platform for experimentation with both biology-inspired and machine-learning-inspired learning approaches. We will present experiments supporting these assertions in Sections 3.2 and 3.3. From a system design perspective, we see these results as evidence that the design is suitable as a unit of scale for a large-scale accelerated neuromorphic learning architecture.

Finally, we will discuss related work and give an outlook on future developments in Section 5.

2. THE BRAINSCALES-2 SYSTEM

2.1. System Architecture

This section will give an overall description of the BrainScaleS-2 architecture in terms of its constituting components. We have taped out several scaled-down prototype versions and now successfully commissioned a full-scale single-core system. This single-core system can serve as the unit of scale for larger-scale designs involving multiple neuromorphic cores. In **Figure 1** we show an overview of the system architecture.

A single neuromorphic BrainScaleS-2 core consists of a full-custom analog core combining a synaptic crossbar, neuron circuits, analog parameter storage, two digital control- and plasticity-processors, and the event routing network responsible for spike communication. The physical design is divided into four quadrants, each featuring a synaptic crossbar with 256 rows and 128 columns. The neuron circuitry is also partitioned in this way, with each of the 512 neuron circuits associated with one column. The two digital processors (Friedmann et al., 2017) located at the top and bottom of the design are responsible for the analog core's upper and lower half, respectively. Using single-instruction multiple data (SIMD) vector extensions, they can read and write the digital state of their half of the synaptic crossbars row-wise in parallel and readout analog traces via a 512-channel column analog-to-digital converter (CADC) (there are 256 channels per quadrant, pairwise associated to correlation and anti-correlation measurement). In Section 2.3 we will give a more detailed description of the digital processors' role in the realization of plasticity. The event routing network takes up the cross-shaped space dividing the four quadrants and is extendable in all directions. Operating on address event-encoded packets,

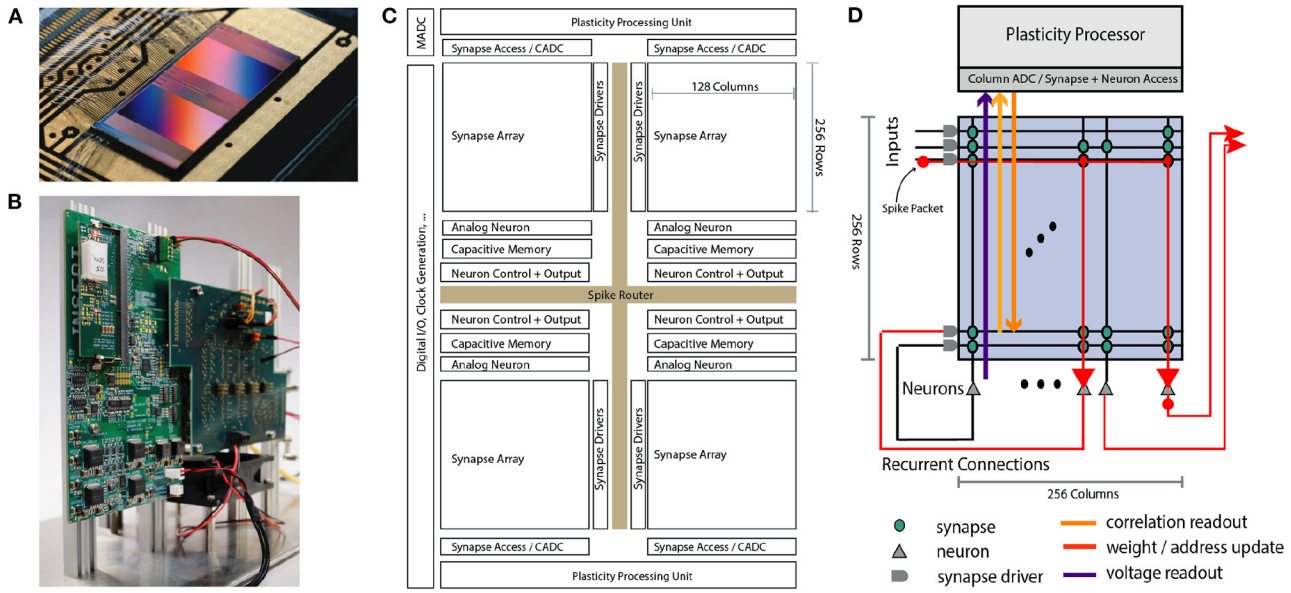

FIGURE 1 | Overview of the BrainScaleS-2 System architecture. **(A)** Bonded chip on its carrier board, one can see the two synaptic crossbar arrays. **(B)** Test setup, with the chip (covered by white plastic) mounted on a carrier board. The FPGA and I/O boards have been designed by our collaboration partners at TU Dresden. **(C)** Schematic floorplan of the chip: Two processor cores with access to the synaptic crossbar array are on the top and bottom. The 512 neuron circuits and analog parameter storage are arranged in the middle. The event router routes events generated by the neurons and external events to the synapse drivers and to/from the digital I/O located on the left edge of the chip. **(D)** Conceptual view of the system architecture in spike processing mode: Event packets (red dot) get injected by the synapse driver into the synaptic crossbar, where they cause synaptic input integration to occur in synapses with matching addresses (indicated by red lines). Membrane voltage accumulation eventually results in spike generation in the associated neuron circuits. The resulting spikes are routable to both synapse drivers or external output. The plasticity processing unit has low latency and massively parallel access to synaptic weights, addresses, correlation measurements, and neuron membrane voltage dynamics during operation. Plasticity rules and other learning algorithms can use these observables to modify all parameters determining network emulation in an online fashion.

it connects the digital output of neuron circuits, external input, and on-chip Poisson sources and routes them to synaptic rows of one of the quadrants or off-chip targets. The digital interface of events and data to an external system implements a custom hardware link protocol (Karasenko, 2020), which supports both reliable data transfer and efficient event communication.

The single-core BrainScaleS-2 system has been integrated into two hardware setups so far: One system tailored for commissioning and analog measurements (shown in **Figure 1B**). In addition, we designed and implemented a fully integrated "mobile" system meant for edge deployment, which incorporates a Zynq FPGA with embedded ARM cores that can run Linux and the BrainScaleS-2 neuromorphic core in a small form-factor (Stradmann et al., 2021). It, therefore, can be deployed entirely independently from any external host. In both cases the FPGA is used for real-time control, buffering of external stimulus and output data. It also manages external memory access for the plasticity processors and provides connection from and to the host system.

Beyond these two hardware setups, we also see the single-core system as a *unit of scale* for large-scale neuromorphic architectures. The most immediate step that does not require a modified ASIC architecture is integrating multiple single-core chips into a larger system by implementing an external event-routing architecture. Work in this direction is underway. We can also increase the reticle size and extend the on-chip digital routing architecture appropriately. With this approach, about 8 cores could be integrated on a single reticle. We will discuss some future directions in Section 5.

Further improvements might consider modifications to the core architecture itself. Future revisions will include on-chip memory controllers, thereby eliminating the need to rely on an external FPGA. The overall modularity of the architecture would also allow swapping out the synaptic crossbar, neuron circuit, or plasticity processor implementation. The neuron circuit implementation underwent several iterations until it converged on the design reported here. We will turn to this aspect of the system next and then describe the plasticity and control processor in more detail.

2.2. Accelerated Analog Emulation of Neural Dynamics

The emulation of neuron and synapse dynamics takes place in dedicated mixed-signal circuits, which—in combination with other full-custom components—constitute the analog neuromorphic core. A full-sized application-specific integrated circuit (ASIC) encompasses 512 neuron compartments with versatile and rich dynamics. They evolve at 1.000-fold accelerated time scales compared to the biological time domain, paying

tribute to the characteristic time constants of the semiconductor substrate. In its core, the neuron circuits faithfully implement the adaptive exponential integrate-and-fire (AdEx) model (Brette and Gerstner, 2005),

$$C_\mathrm{m}\dot{V} = -g_\mathrm{l}(V - E_\mathrm{l}) + g_\mathrm{l}\Delta_\mathrm{T}\exp\left(\frac{V - V_\mathrm{T}}{\Delta_\mathrm{T}}\right) - w + I\,, \quad (1)$$

$$\tau_\mathrm{w}\dot{w} = a(V - E_\mathrm{L}) - w\,, \quad (2)$$

where the first differential equation describes the evolution of the membrane potential V on capacitance C_m. The membrane accumulates currents $I = I_\mathrm{syn} + I_\mathrm{stim}$, which encompass direct external stimuli as well as currents originating from synaptic interaction. Furthermore, g_l represents the leak conductance pulling the membrane toward the leak potential E_l. The exponential term implements the strong positive feedback emulating the coarse shape of the action potential of a biological neuron and is controlled by the exponential slope Δ_T and the soft threshold V_T. An outgoing spike is released as soon as the membrane potential crosses the hard firing threshold V_th. In that case, the membrane is clamped to the reset potential V_r and held there for the refractory period t_r. A second differential equation captures the dynamics of the adaptation current w allowing the neuron to adapt to its previous activation and firing activity. The adaptation state decays back to zero with a time constant τ_w and is driven by the deflection of the membrane potential, scaled with the subthreshold adaptation strength a. In case of an action potential, w is incremented by b implementing spike-triggered adaptation. A more detailed, transistor-level description of a previous version of the BrainScaleS-2 neuron circuit can be found in Aamir et al. (2018b).

Each neuron circuit can be configured individually via 80 bit of local configuration static random access memory (SRAM) as well as 24 analog parameters which are provided by an on-chip digital-to-analog converter (DAC) with 10 bit resolution (Hock et al., 2013). The analog parameters allow to control all potentials and conductances mentioned in Equation (1) for each neuron individually—the model dynamics can therefore be tuned precisely and production-induced fixed-pattern deviations can be compensated. This in particular allows to calibrate each circuit to a specific set of model parameters, which may either be homogeneous across the whole array or custom to individual neuron instances. Other aspects of the neuron, such as the refractory time t_r or the membrane capacitance C_m, can be directly configured via the locally stored digital configuration, which can also be used to en- or disable certain features of the neuron. For example, the adaptation current as well as the exponential term in Equation (1) can be disconnected from the membrane, reducing the AdEx model to the simpler leaky integrate-and-fire (LIF) neuron model. When disabling also the leak and threshold circuits, the neuron can be employed to linearly accumulate charges and therefore—in conjunction with the synapse array— implement analog matrix multiplication (see Section 3.3.2).

On the other hand, the neuronal dynamics can also be extended. Using additional resistors and switches between the neuron circuits, larger cells with an increased synaptic fan-in as well as intricately structured neurons can be formed (Aamir et al., 2018a; Kaiser et al., 2021).

Each neuron circuit integrates synaptic stimuli from a column of 256 plastic synapses. BrainScaleS-2, in particular, features time-continuous current- as well as conductance-based synapses with exponentially decaying kernels. The total current

$$I_\mathrm{syn} = \sum_i w_i S_i(t) * \mathrm{e}^{-t/\tau_\mathrm{syn}} \quad (3)$$

hence results from the sum over all associated synapses i, their respective weights w_i, the presynaptic spike trains $S_i(t) = \sum_j \delta(t - t_j)$, and the synaptic time constants τ_syn, which can be chosen independently for excitatory and inhibitory stimuli. On BrainScaleS-2, the weights are stored locally per synapse in 6 bit SRAMs and modulate the amplitude of an emitted current pulse. Along them, each synapse also holds a 6 bit source address which is compared to the label of afferent events and lets the synapse then only responds to matching stimuli. Hence, the network structure is determined not only through the digital event routing network but also by synapse-local properties. Synapse addresses, in particular, allows to map sparse networks efficiently and change the connectome by inserting and removing synaptic connections dynamically (Billaudelle et al., 2021).

Spike timing-dependent plasticity (STDP) and related correlation-based plasticity rules are supported through analog sensor circuits within each synapse (Friedmann et al., 2017). They continuously measure the exponentially decaying pair-wise correlation between post- and presynaptic spikes and accumulate them as an observable for weight update calculations (cf. Section 2.3). In addition, BrainScaleS-2 supports a presynaptic modulation of events, which is exploitable for the implementation of short-term plasticity (STP) (Bi and Poo, 1998) and allows to inject graded spikes. By combining the latter with neurons with disabled spiking dynamics—hence acting as simple integrator circuits—BrainScaleS-2 also supports the execution of non-time-continuous vector-matrix multiplications (see Section 3.3.2).

2.3. Hybrid Plasticity and Versatile Digital Control

Besides the accelerated and faithful emulation of neuron and synapse dynamics realized by the analog neuro-synaptic core described in the previous section, the system features two digital plasticity and control processors (Friedmann, 2013), which we refer to as plasticity processing unit (PPU). The overarching goal of this part of the system is to complement the flexible and configurable neuron and synapse architecture with an equally flexible digital control architecture. Here we highlight several use cases of this design choice: implementation of programmable hybrid plasticity, automatic on-chip calibration, parallel readout of analog observables for in-the-loop learning, orchestration of analog artificial neural network computation, and simulation of virtual environments.

Apart from speed, a big problem of physical implementations is their limited flexibility, especially regarding learning rules. BrainScaleS-2 uses a "hybrid plasticity" scheme (Friedmann et al.,

2017), combining analog measurements with digital calculations to increase the flexibility while keeping the advantages of an accelerated physical model, like simultaneously observing all correlations between pre-and postsynaptic signals. In a moderate-sized network, this amounts to tens of thousands of measurements per second for each synapse. Thus, a single BrainScaleS ASIC can perform several tera-correlation measurements per second. Compared to biological model dynamics, the emulation runs one thousand times faster; a speedup factor that, even for small to medium-sized plastic networks, is typically out of reach to software simulations (Zenke and Gerstner, 2014).

To implement the plasticity rules themselves, i.e., to calculate new weights, topology information, and neuron parameters during the continuous-time operation of the network, the hybrid plasticity architecture relies on specialized build-in SIMD units in the two microprocessor cores. They interface directly with the synaptic crossbar and the neuron circuits via the CADC, which, as the name suggests, can perform simultaneous measurements of analog quantities in one row and across all 256 columns of the synaptic crossbar. The result of the correlation measurements can then enter plasticity programs, which can perform both fixed-point and integer arithmetic operations on vectors of either 128×8 bit or 64×16 bit entries. The weights and addresses stored in the synaptic array and the voltage traces from the neuron circuits allow for flexible plasticity computations bridging multiple timescales. In particular, we have demonstrated several versions of spike timing-dependent plasticity (STDP)-based learning rules (R-STDP Wunderlich et al., 2019, homeostatic plasticity Cramer et al., 2020a), as well as learning rules that compute updates based on small artificial neural networks (Bohnstingl et al., 2019) and structural plasticity (Billaudelle et al., 2021).

The scalar part of the processor core can operate independently and take responsibility for scheduling the data-parallel instructions, data transfers, and measurements. Scoreboards track data dependencies in an in-order issue out-of-order retire scheme, which allows both the vector unit and the scalar unit to perform, for instance, arithmetic operations, that are independent of the completion of potentially higher latency load/store operations or CADC measurements, thereby reducing overall execution time. This asynchronous operation of vector and scalar units is beneficial, in particular, for plasticity programs.

Besides plasticity programs, the parallel access to the analog synapse and neuron state is also helpful in the gradient-based learning approaches discussed below. The surrogate gradient-based learning approach relies on CADC samples of the membrane voltage during experiment execution. A program running on the PPU performs this sampling in a tight loop and then writes the resulting traces to either internal SRAM or external memory. The external bandwidth of the system and the DDR3-memory the FPGA interfaces with make this approach feasible. During artificial neural network in-the-loop training and inference, the processor cores perform data transfer, accumulation of partial results, and the analog readout of matrix-vector multiplication results.

As will be discussed in the following Section 3.1, the neuron circuits can be adjusted by calibration of 24 parameters each. In addition, the parallel access to the neuron circuit membrane dynamics via the CADC allows us to implement efficient on-chip calibration, as done in several of the experiments reported here.

Finally, the processor cores can simulate virtual environments. This use case is enabled because the processor cores can inject spikes into the synaptic crossbar and readout rates and voltage traces from the neuron circuits. It is, therefore, possible to close the agent-environment loop by simulating the agent's state and the environment interaction on the PPU and implement the agent's action selection (in part) as a spiking neural network. We will give a detailed description of one such use case in Section 3.2.1.

The flexibility of the digital architecture is in large part also enabled by a compiler, the C++ programming language, and library support. The scalar processor cores implement a subset of the 32 bit POWER instruction set architecture (PowerISA, 2010). The SIMD vector instructions are custom but generally follow the conventions of the VMX SIMD instruction extension (PowerISA, 2010). We have modified the GCC compiler toolchain to support the custom vector instructions based on the pre-existing support of the POWER instruction set architecture. With this compiler support as a foundation, we have implemented support for the C++ standard library and ensured that our hardware abstraction library (HAL) is usable both on the embedded processor cores and on the host system, with a user-transparent change of hardware access modes. Details were reported by Müller et al. (2020). Finally, we also provide abstractions for scheduling and executing plasticity rules.

While the on-chip memory resources are limited to 16 KiB SRAM per processor core, the cores have access to an external memory interface, which can back both the 4 KiB per core instruction cache and provide higher-latency access to FPGA attached memory or block RAM. This memory architecture allows for both the execution of latency-critical code without external memory access and less latency sensitive but memory intensive setup and calibration code. Future revisions will include on-chip memory controllers for direct access to suitable external memory technology.

3. APPLICATIONS OF THE BRAINSCALES-2 SYSTEM

3.1. Faithful Emulation of Complex Neuron Dynamics

Analog neuromorphic systems can usually suffer from temporal noise, fixed-pattern parameter deviations, and a divergence from the original model equations. As elaborated in Section 2.2, BrainScaleS-2 goes a long way to accomplish an extensive and at the same time detailed control over each individual circuit and thus model parameter.

We employ calibration to find configuration parameters for each circuit such that its observable characteristics match a given target. Due to the device-specific nature of fixed-pattern deviations, this calibration is an iterative process

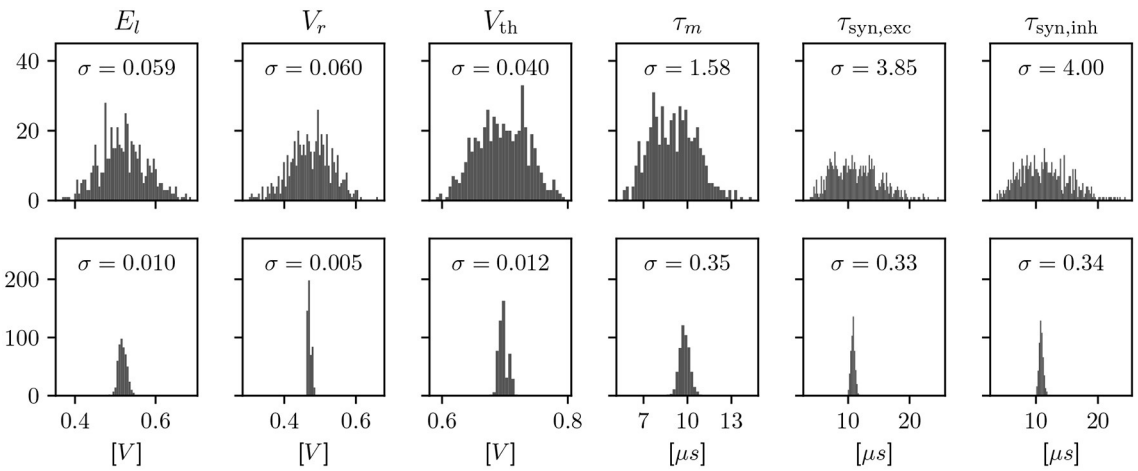

FIGURE 2 | Histograms showing a characterization of LIF properties of neurons before (top) and after (bottom) calibration. Each histogram shows all 512 neuron circuits on a single ASIC. In the top row, the configuration has been set equal for all neurons, to the median of the calibrated parameters. This results in different model characteristics due to device-specific fixed-pattern noise arising during the manufacturing process. After calibration, the analog parameters, such as bias currents and voltages, are selected such that the observed characteristics match a target.

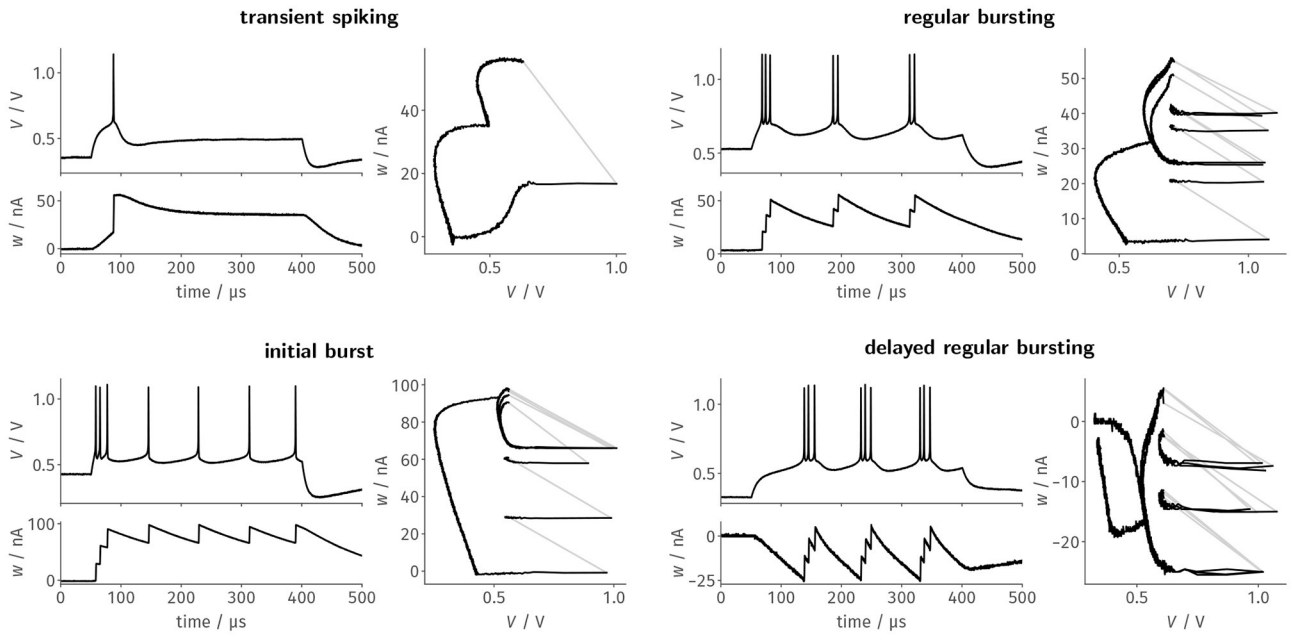

FIGURE 3 | Faithfully emulating the original AdEx equations, BrainScaleS-2's neuron circuits can be configured to generate distinct firing patterns as a response to a constant current stimulus. Here, we tuned the neuron circuits exemplarily to replicate four of the patterns described by Naud et al. (2008) using automated calibration routines. Each of the four panels features the time evolution of the membrane trace and the adaptation current, as well as the resulting trajectory through the phase space.

involving a measurement on each of the specific circuits. These measurements are a minimal hardware experiment, typically based around an ADC or spike rate measurement, in order to characterize one observable at a time. The effects of calibration are visualized for LIF neurons in **Figure 2**. However, the scope of calibration extends beyond those, to, e.g., the AdEx model, multicompartment functionality, and technical parameters that don't correspond to a term in a model, but are necessary for the circuitry to behave as expected.

In this section, we want to use this high configurability to replicate the original firing patterns analyzed by Naud et al. (2008) and to show how the presented system can be used to emulate multi-compartmental neuron models. For these experiments, system configuration as well as stimuli data are

generated on a host computer and then transferred on the FPGA. The FPGA handles experiment control and buffers recorded voltages. These voltage recordings are performed with the fast on-chip analog-to-digital converter (ADC) which offers a resolution of 10 bit and a sampling frequency of about 29 MHz. At the end of the experiment, the voltage recordings are transferred to the host computer and evaluated.

3.1.1. Replication of Biological Firing Patterns

In order to test the full AdEx model as depicted above we selected sets of model parameters from Naud et al. (2008) and mapped all voltages, currents, and conductances to the circuit's native domain. Specifically, we honored the acceleration factor of 1.000, the physical membrane capacitance, and the voltage range of the silicon neuron. We then tuned our neurons to these model parameters by utilizing automated calibration routines.

For each of the designated firing patterns, we then stimulated the neurons with a respective step current $I_{\text{stim}} = I_0 \cdot \Theta(t - 50\,\mu\text{s}) \cdot \Theta(400\,\mu\text{s} - t)$. We recorded the membrane potential as well as the adaptation state voltage and— based on the latter— estimated the adaptation current flowing onto the membrane. **Figure 3** shows these two resulting traces as well as the resulting trajectory through phase space for a single neuron. The four exemplarily chosen firing patterns highlight different aspects of the neuron design: *Transient spiking* requires both large spike-triggered adaptation increments b and a strong subthreshold adaptation a to emit a single spike as a response to the stimulus onset and remain silent for the remainder of the current pulse. *Regular bursting* and the *initial burst*, in contrast, already emerge for configurations with small a and mainly rely on spike-triggered adaptation. These two patterns primarily differentiate themselves in the exact choice of the reset potential V_{r} in relation to the threshold voltage V_{T}, which demonstrates the precise control over the respective circuit parameters. Finally, *delayed regular bursting* relies on an inverted subthreshold adaptation ($a < 0$) leading to positive feedback.

For the last three patterns, the trajectory through the phase plane spanned by V and w nicely demonstrates the precise and reproducible dynamics of the circuit, which are especially highlighted by the remarkable stability of the limit cycles for the periodic spiking activity during stimulation. While here only shown for a single neuron, all patterns could—without manual intervention—be reliably reproduced by most, if not all, of the neuron circuits. For example, regular bursting could be simultaneously configured for all of the 128 tested neurons.

3.1.2. Multi-Compartmental Neuron Models

To further demonstrate the high configurability of the system at hand we want to show how multi-compartmental models can be emulated. As mentioned in Section 2.2, BrainScales-2 offers the possibility to connect several neuron circuits and therefore allows to implement various compartmental neuron models (Kaiser et al., 2021).

A passive compartment chain model can for example be used to replicate the behavior of a passive dendrite, **Figure 4A**. We inject synaptic input in one of the compartments and investigate how the excitatory post synaptic potential (EPSP) travels along the chain of compartments. For that purpose, we once again use the fast on-chip ADC to record the membrane potentials in the different compartments. In the first row the input is injected into the left compartment. As expected, the EPSP becomes smaller and broader as it travels along the chain. The extent of the attenuation can be controlled by the conductance between the different compartments. When looking at the case where the input is injected into the second compartment the influence of the neuron morphology becomes obvious. The height of the EPSP is smaller as compared to the injection in the first compartment. This is due to the lower input conductance: the second compartment has two neighbors as compared to the single neighbor for compartments at the end of the chain.

Dendrites are not simple passive cables but are able to initiate local regenerative events (Larkum et al., 1999; Schiller et al., 2000; Major et al., 2013). On BrainScaleS-2 each compartment is made up of one or more fully functional neuron circuits and can therefore replicate dendritic spikes. Besides sodium-likes spikes, which are modeled by the AdEx model, the neuron also supports plateau-like spikes. **Figure 4B** illustrates the model of a dendritic branch which splits into two thinner dendrites. The first compartment, *compartment 0*, of one of the branches is configured to initiate plateau potentials. We inject synaptic inputs in two compartments with a fixed delay between them and record the membrane potential at the spike initiation zone. As expected, spiking depends on the spatio-temporal distribution of the inputs (Williams and Stuart, 2002; Polsky et al., 2004). While inputs near the spike initiation zone elicit a dendritic spike, more distal inputs fail to cause a threshold crossing. Furthermore, a spike is more easily triggered if the distal input precedes the input at the initiation zone.

3.2. Biology-Inspired Learning Approaches

One underlying goal of the system is to enable the exploration of biologically plausible learning rules at accelerated time scales relative to biology. For our purposes, we consider a learning rule or algorithm to be biologically plausible if it satisfies several criteria. The algorithm should be *spatially* and *temporally* local. By spatially local, we mean that parameter changes computed by the algorithm should rely only on observations that can be locally made at each neuron and synapse. Some aspects of biological plausibility are enforced by the system design itself. The PPUs—coupled to the correlation circuitry implemented in each synapse—facilitate the implementation of such spatially local algorithms, as we have discussed in Section 2.3. A temporally local algorithm should not rely on complete traces of activity but sparse temporal observations.

Here we highlight experiments that have effectively used the underlying hardware capabilities to demonstrate aspects of biologically plausible learning. One aspect of biological learning systems is that they typically interact with an environment. We have realized tasks on the BrainScaleS-2 system in which an agent interacts both with a simulated and physical environment. Tasks range from simple Markov decision processes like maze navigation and Multi-Armed Bandits (Bohnstingl et al., 2019), playing a version of Pong (Wunderlich et al., 2019), to insect navigation and control of an accelerated robot (Schreiber,

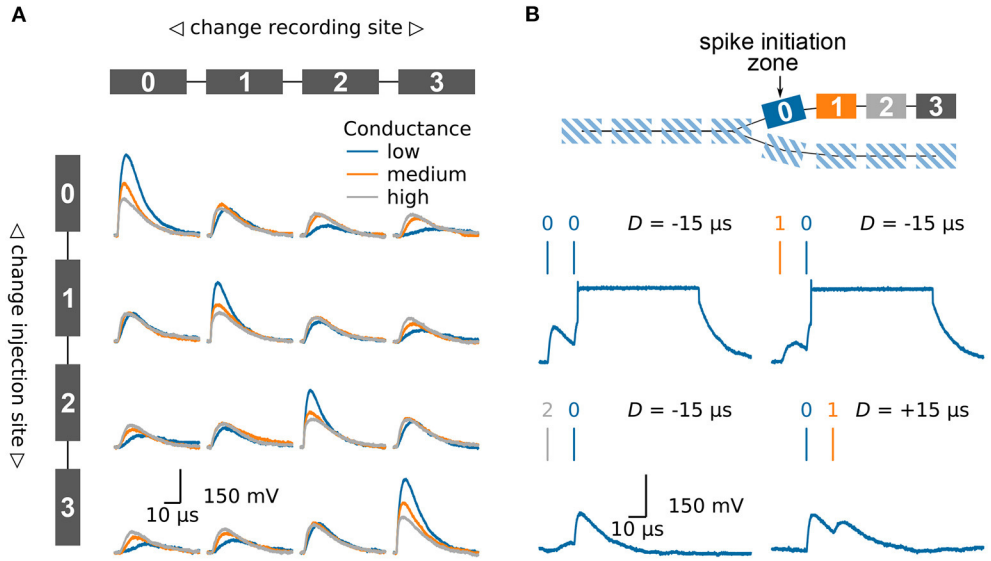

FIGURE 4 | Emulating multi-compartmental neuron models on BrainScaleS-2. **(A)** Four compartments are connected to form a chain, compare model at the top and left side. The traces show the membrane potential in the four different compartments as a synaptic input is injected in one compartment after another. Thanks to the high configurability of the BrainScaleS-2 system, compare Section 2.2, the conductance between the individual compartments can be altered to control the attenuation of the signal along the chain. **(B)** Model of a chain which splits in two. The BrainScaleS-2 system supports dendritic spikes, here demonstrated in the form of plateau potentials. Inputs are injected in different compartments with a fixed delay between them; this is indicated by the vertical bars above the membrane traces. Depending on the spatio-temporal distribution of the inputs a dendritic spike can be elicited. The traces show the membrane potential in *compartment 0*. Figure adapted from Kaiser et al. (2021).

2021). In all of these instances, the rapid reconfigurability and experiment execution time lead to a significant speedup over a simulation on commodity hardware.

The virtual environment and the agents are simulated on the plasticity processor, which can guarantee low latency due to its tight coupling to the analog neuromorphic core. In fact, this lack of deterministic low latency coupling made such experiments difficult on the BrainScaleS-1 systems, where that problem was further emphasized by the additionally increased acceleration factor.

Besides the three reinforcement learning related experiments, we also explored other aspects of biologically plausible learning. For example, the parallel access of the plasticity processing unit to both the (anti-)correlation sensor readings and the digital weight and address settings of the synapse array suggests experiments based on synaptic rewiring and pruning (Billaudelle et al., 2021). This work makes use of the ability of our synaptic crossbar to realize sparse connectivity, as each synapse has a local receptive field of 64 potential inputs. Last but not least, we have performed work exploring criticality and collective dynamics of spiking neurons subject to homeostatic plasticity rules on both the scaled-down prototype systems (Cramer et al., 2020a) and in ongoing work, which we will report on in Section 3.2.3.

3.2.1. Insect-Inspired Navigation

One way to study neural computation is to focus on small functional circuits. While the scale of the single-core system with its 512 neuron circuits and the prototype systems with 32 neurons do not allow the exploration of large-scale dynamics, they should enable the study of functional circuits over long periods. Here we focus on a recently published anatomically constrained model of the path integration abilities of the bee brain (Stone et al., 2017). Path integration enables bees to return to their nest after foraging for food successfully. In adopting the model to the hardware constraints, we had to both translate the rate-based model to a spike-based model and find a way to implement the integration primitives in terms of the plasticity mechanisms available in hardware. The resulting model is scaled-down relative to the model proposed by Stone et al. (2017) to fit the hardware constraints of the prototype system. Furthermore, the model can be run at the 1.000× accelerated time scale relative to biology on a prototype of the BrainScaleS-2 system and, more recently, on the full-scale single-chip system.

The model is evaluated on a task divided into three parts. In the first "foraging" part, the bee performs a random walk starting at the "nest" location, with two sensory neurons receiving light direction information. In a second "return" phase output of the motor neurons is used to determine the movement of the bee back to the nest. Finally, during a "looping" phase, the bee is supposed to remain as close as possible to the nest until the experiment is terminated.

Details of the signal flow and an activity trace are given in **Figure 5A**. A "foraging," "return" and "looping" episode was simulated in 200 ms, therefore corresponds to 200 s in biological time. The virtual environment of the bee, as well as simulating the bee's "foraging" random walk, position, heading

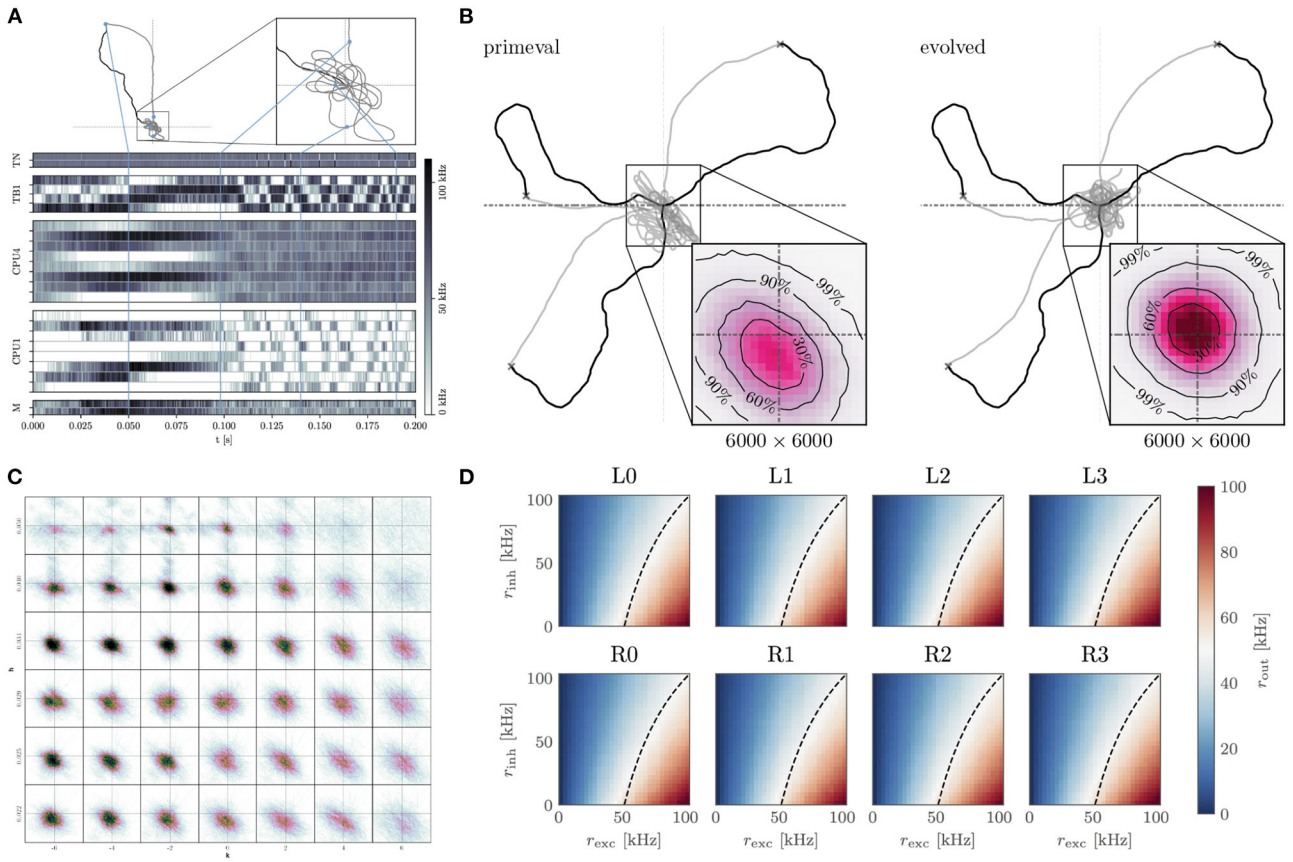

FIGURE 5 | Insect-inspired path integration and navigation on the BrainScaleS-2 accelerated neuromorphic substrate. **(A)** Network activity and path during a single experiment instance. The simulated agent first randomly spreads out from the "nest" until it finds "food." It then returns to the nest using the integrated path information. Once returned, it starts looping around the nest until the experiment is terminated. The blue lines indicate how points along the trajectory correspond to the activity trace shown below. From top to bottom, the neuron activity traces are that of the two sensory neurons (TN), "compass" neurons (TB), integrators (CPU4), steering (CPU1), and motor neurons (M). Signals flow from TN → TB, TB → CPU4, CPU1; CPU4 → CP1 and CP1 → M. Details of the network architecture were given by Schreiber (2021). **(B)** We performed evolutionary optimization of the weights controlling the behavior of our agents. Here we show 3 sample trajectories each of the initial (primeval) and evolved population. Inset is a histogram over 1.000 trajectories in the final "looping" phase of the evaluation, zoomed in to a 6.000 × 6.000 square of positions at the origin. As can be seen, the evolved population reaches a more symmetrical and tighter looping behavior. **(C)** Influence of two hyperparameters h, k on the integration performance of the spiking integrators, each 12.000 × 12 000 square of positions contains a histogram of 100 trajectories in the looping phase ($t > 2t_{\text{return}}$) for a total of 4.200 trajectories. **(D)** Response to excitatory and inhibitory input rates of the calibrated CPU1 neurons. The dashed line indicates where 50 % of the maximum output rate is expected. Calibration of these neurons, as well as all the other neurons, was done *on-chip* using the PPU, in the case of the implementation done on the full-scale BrainScaleS-2 system. **(A–D)** are adapted from Schreiber (2021), **(D)** uses data from Leibfried (2021).

direction in the environment, and sensory perception, are implemented on the PPU. It, therefore, closes the perception-action loop in conjunction with the accelerated emulation of the path integration circuit on the neuromorphic core. Moreover, the behavior of the CP4 integrator neurons relies on weight modifications of axoaxonic synapses, which are implementable due to the flexible synaptic modifications that can be performed by the PPU.

The acceleration factor also allows us to efficiently sweep the hyper-parameters of the integrator neurons (**Figure 5C**). Moreover, we are able to calibrate the neurons for this task using on-chip resources (**Figure 5D**). Finally, we performed an evolutionary optimization in order to improve the agents' distance to the nest during the looping phase (**Figure 5B**). Without the acceleration factor, the hyperparameter sweep would have required 9.6 day. Instead, it can be performed in approximately 14 min. Similarly the evolutionary optimization over 200 epochs of 1.000 individuals would have required 448 day, instead it could be completed in roughly 12 h. The output data required by the evolutionary algorithm is buffered by the FPGA and evaluated on the host. The host is also responsible for the initial and all subsequent configurations of the system, as well as the implementation of the evolutionary algorithm.

Therefore, we believe this to be a compelling case study of the system's modeling capabilities concerning small embodied neuromorphic agents and of how contemporary research in computational neuroscience could benefit from an accelerated physical emulation platform to evaluate experiments. While implementing such a model on neuromorphic hardware enforces constraints on the modeler not present in a free-form

von-Neumann modeling approach, it also has benefits. On a technical side, the acceleration factor is guaranteed even when scaling to larger circuits, including plastic synapses. On the conceptual side, many of the constraints present in the hardware, namely spike-based communication and local plasticity rules, are also believed to be present in biology. Therefore, one is encouraged to fully commit to a modeling approach without any shortcuts afforded by using a von-Neumann architecture to simulate the model. Furthermore, the experiment demonstrates that it is possible to calibrate the neuron circuits with on-chip resources, which is crucial for a scalable and host-independent calibration.

A detailed description of the implemented spike-based model and the experiments, including the evolutionary optimization, was given by Schreiber (2021) and will be further elaborated on in a forthcoming publication (Schreiber et al., 2022).

3.2.2. Accelerated Closed Loop Robotics

We can also consider another family of applications: Using the BrainScaleS-2 system to control another physical system. This presents several challenges: The accelerated neuron dynamics means that the natural time scale at which the system could interact with a real environment would be on the order of microseconds. Similarly, sensor information to be processed by the system needs to be on this time scale, at least if spikes are meant to be used as the information processing primitives. As a case study, Schreiber (2021) implemented an accelerated mechanical system consisting of two actuators moving a light sensor over an illuminated surface or screen. Here we give a brief overview and refer to Schreiber (2021) for a detailed exposition. While this robotic design has limited practical purpose, it illustrates the key challenges any such attempt faces. This begins with the challenge of translating signals between the robot and spiking domain, where the challenge is to interpret the sensor reading as spikes naturally and to convert the "motor" neuron output into actuator input. Here, we solve this in two different ways: In a first prototype, the sensor to spike conversion was done by connecting modular analog circuits, featuring circuits for differentiation, inversion, adder, noise, and spike generation, which ultimately could produce spike input to a parallel FPGA interface. While this came closest to the ideal of analog and spike-based communication, we replaced this component with a fully digital micro-controller-based solution in a second iteration. This implementation choice is beneficial because other experiments could also use this microcontroller to implement more sophisticated virtual environments. In both versions of the design, spikes of the motor neurons with a width of roughly 500 ns were converted into actuator input by using pulse-shapers. This required motor drivers and actuators, which could meaningfully react to input pulses with approximately 10 microsecond duration. Fortunately, voice coil actuators, as used in commodity hard drives, precisely have this property. **Figure 6B**, shows an overview of the signal path. The actuators move a sensor over the illuminated surface or screen. In **Figure 6** we show task performance and sensor trajectories of a "maximum finder," that is, the task of the agent is to find the local intensity maximum following the light intensity gradient. And indeed, the following of (chemical) gradients is one way in which biological organisms find food sources. The accelerated nature of the experiment execution allows for fast evolutionary optimization of the required weight matrix (see **Figures 6D,E**). Other experiments demonstrate that, for example, one can construct networks capable of rapidly detecting light intensity "edges."

Beyond the experiments done so far, it would also be possible to display a maze environment on the screen and let the neuromorphic agent physically solve the maze, with rewards being encoded either virtually by the micro-controller or by light intensity changes in the maze. Although initial calculations suggested that the speed would be sufficient to produce light intensity changes at the right time scale, ultimately, we were only able to achieve regulation oscillations with a period of approximately 10 ms. While this is still quite fast, it does not naturally correspond to the time scale of the system. In some sense, the neuromorphic system, therefore, can only control this environment in slow motion. This is less of an issue for a maze-like environment, where the decision procedure could be more abstractly interpreted in terms of "moves."

Besides the initial configuration and the implementation of the evolutionary optimization, which requires recorded data buffered on the FPGA and transferred to the host, the experiment is implemented in a host independent fashion. It therefore demonstrates accelerated closed-loop neuromorphic computation.

Overall we believe using our accelerated neuromorphic system for rapid spike-based control and sensing has many exciting future applications. Further experiments can extend the proof of principle we present here in several directions. For example, instead of voice coils, one could control piezo-electrically actuated mirrors or lenses to divert laser light in a physical experiment or control micro-sized or high-speed aerial vehicles. Other high-speed applications could include motor control circuits and ultra-sound or radar applications, particularly involving active sensing or phased arrays.

3.2.3. Collective Dynamics

One way to adapt recurrent spiking neural networks (SNNs) to perform information processing is to deliberately exploit collective dynamics. Particularly promising are the dynamics emerging at a so-called *critical point* at which systems fundamentally change their overall characteristics, transitioning between e. g. order and chaos or stability and instability. Being at this point, systems maximize a set of computational properties like sensitivity, dynamic range, correlation length, information transfer and susceptibility (Harris, 2002; Barnett et al., 2013; Tkačik et al., 2015; Munoz, 2018).

Here, we showcase the tuning of plastic recurrent SNNs to and away from criticality by adapting the input strength K_{ext} on a prototype of the BrainScaleS-2 system (Cramer et al., 2020a). The CADC as well as the PPU facilitate an on-chip implementation of the STDP-based synaptic plasticity required to adapt the collective dynamics of our SNNs. Within these experiments, the FPGA is only used for experiment control as well as spike injection. On the latest chip revision, the latter

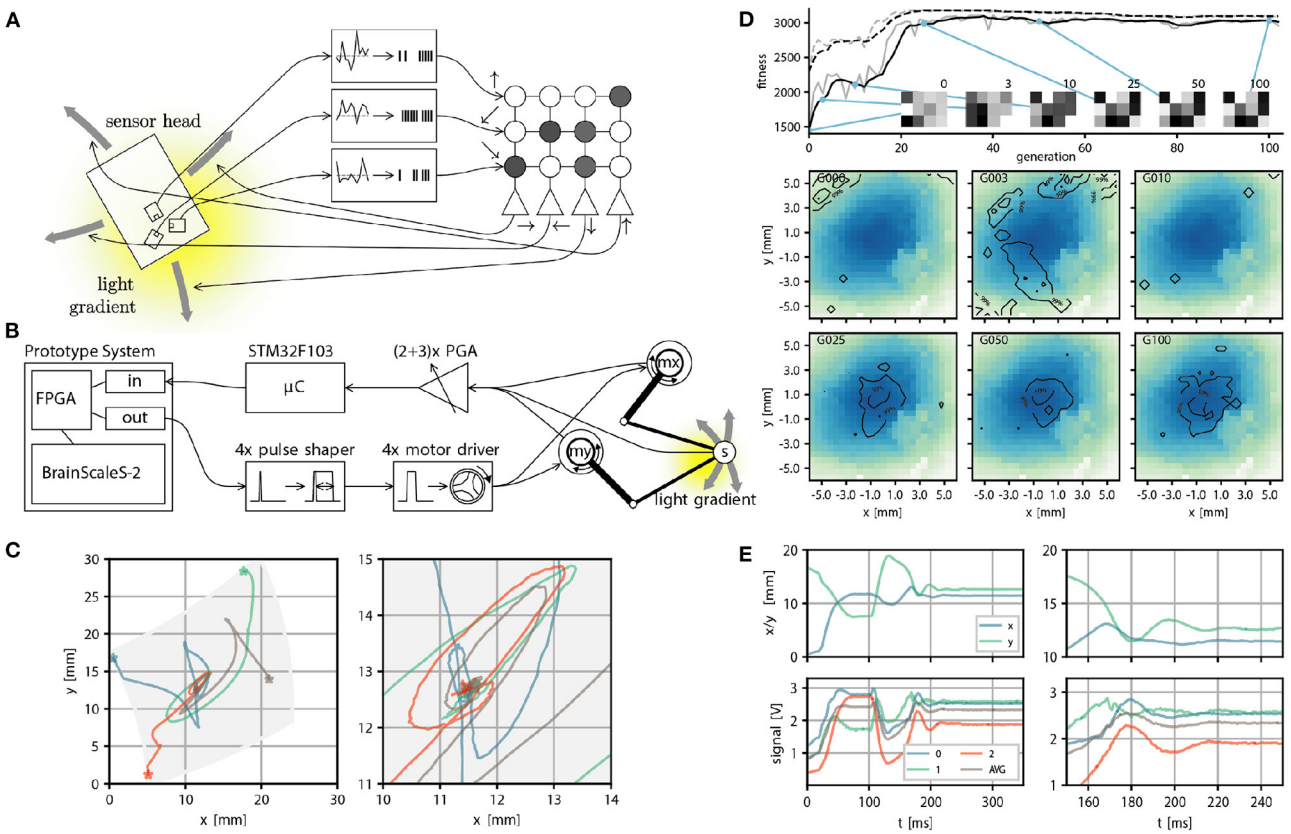

FIGURE 6 | Real world closed loop interaction with the BrainScaleS-2 system. **(A)** Conceptual overview of the example experiment, a sensor head with three light sensors is moved over an illuminated surface. Sensor readings are converted into spike input to 3 input sources in a small feedforward spiking neural network. Spike output produced by 4 neurons is converted into motor commands, which move the sensor head on the surface. The goal is to follow the light gradient. **(B)** Physical realization of the example experiment from right to left: The PlayPen2 consists of two actuated arms which pantographically move a sensor head over a screen or illuminated surface. Signals from the sensor head are digitally processed by a micro controller and converted into spikes send into a FPGA used to interface with a scaled down BrainScaleS-2 prototype system, which implements the small feedforward spiking network. Spike outputs are routed in the FPGA to the Spike I/O interface and converted by pulse shapers into motor command pulses. **(C)** Example trajectories of the sensor head on the surface, with gray indicating the accessible region (left) and zoom in on the center region where the brightness maximum is (right). **(E)** Position (top) and brightness signals (bottom) of the sensor head over time. The two panels on the left show the full time course. The neural network starts control at $t = 100$ ms and stops at $t = 250$ ms. The two panels on the right show a zoom in on the interval $t \in [100, 200]$ms, with the gray curve indicating an average over all brightness readings. **(D)** We perform evolutionary optimization of the 4×3 weight matrix both from a random initial weight configuration. We show the moving average of the fitness in black and the fitness at a certain generation in gray, both for the top three individuals (dashed line) and the population average (solid line). In addition we display the weight configuration of an arbitrary individual at 6 selected generations (0, 3, 10, 25, 50, and 100). Finally we show a qualitative performance evaluation at the same generations of the average weight matrix over 100 experimental runs divided into 4 starting positions. The contourlines show where 99, 50, and 10% of all trajectories ended after 225 ms, when the maximum brightness should be reached. The contourlines are overlayed over a plot of the brightness sampled from the photo diode signals over all runs and generations. Missing pixels correspond to locations not reached at any time. **(A–E)** were adapted from Schreiber (2021) and **(C)** from Billaudelle et al. (2020).

can be achieved by drawing on the on-chip spike generators of BrainScaleS-2, thereby reducing the strain on I/O. Here, the hybrid plasticity approach in combination with the accelerated nature of the BrainScaleS-2 architecture allows us to fully exploit the associated advantages by bridging the gap in time scales between neuro-synaptic dynamics, network dynamics, plasticity evaluation as well as acquisition of long-lasting experiments for statistical analysis.

With our implementation, we showcase emergent autocorrelation times for low input strengths K_{ext}, significantly exceeding the time scales of single-neuron dynamics (**Figure 7A**). Most notably, adjusting K_{ext} allows us to precisely tune the time scale of the collective dynamics. These dynamics can be deliberately exploited for information processing by tying on the reservoir computing framework (Jaeger, 2001; Maass et al., 2002). To that end, we characterize the interplay of collective dynamics and task complexity by training a linear classifier on the host computer based on the spike trains emitted by the BrainScaleS-2 chip. While long time scales for low K_{ext} only boost the performance for complex memory-intensive tasks, simple tasks profit from short intrinsic time scales (**Figure 7B**). Hence, every task requires its own dynamics to be solved optimally (Cramer et al., 2020a). The required tuning can be realized by dynamically adjusting K_{ext} under the constant action

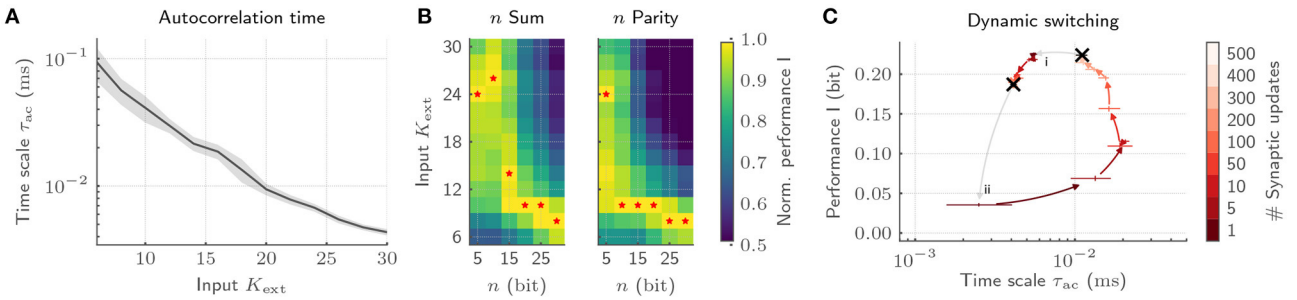

FIGURE 7 | Exploiting collective dynamics for information processing on BrainScaleS-2. **(A)** The autocorrelation time τ_{ac} of recurrent SNNs can be controlled by changing the input strength K_{ext} under the constant application of homeostatic regulation. **(B)** The emerging autocorrelation times for low K_{ext} can be exploited for complex, memory-intensive task processing. Both, a n bit sum as well as a n bit parity task profit from the complex processing capabilities for high n. Low n task, in contrast, profit from short time scales and hence high K_{ext}. The optimal K_{ext} for each n is highlighted by red stars. As a result, each task requires its own dynamic state. **(C)** The adaptation of the network dynamics to task requirements can be achieved by switching the input strength under the constant action of homeostatic regulation irrespective of the initial condition. Here, the transition from a state with long time scales to short ones is completed with only a few homeostatic weight updates (i). The reverse transition requires a longer relaxation phase (ii).

of the synaptic plasticity. This switching can again be efficiently achieved on BrainScaleS-2 by exploiting the acceleration. The transition from a state with high to low autocorrelation time is completed with 50 synaptic updates amounting to only 50 ms, whereas the reverse transition requires 500 updates and hence 500 ms (**Figure 7C**). Moreover, this switching leads to comparable dynamical regimes irrespective of the initial condition and is characterized by a low energetic footprint when drawing on the on-chip spike sources. With this, we provide not only an understanding of how the collective dynamics can be adjusted for efficient information processing, but in addition, showcase how the physical emulation on BrainScaleS-2 allows to bridge the vast range of time scales in the associated experiments which render equivalent implementations on conventional hardware prohibitively expensive.

3.3. Gradient-Based Learning Approaches

Both the individual analog circuits and the overall system can be considered to be parameterized physical systems. A particular task can be represented as a constraint optimization problem involving a loss function and constraints which implement the input-output relation. One approach to such a constraint optimization problem is to estimate gradients of the parameters and subsequently perform some form of (stochastic) gradient descent. Compared to digital computers the analog nature of the core components leads to additional challenges. Whereas, digital neuromorphic systems mainly need to be concerned with the limited precision in their digital arithmetic and otherwise can exactly simulate the operation of their system, this is not the case for analog neuromorphic systems.

Just like with any other parametrized physical system it is important to have a model of its behavior in order to perform this optimization. As a simple example to keep in mind, think of a physical pendulum, such as a ball hanging on a piece of string. The physical parameters of such a pendulum, namely the length of the string L and the mass of the ball m enter any *model* of this system. A good model of a physical system does not necessarily need to capture all the details of the physical situation to be useful. For example for small initial angles the motion of a pendulum is well described by a damped harmonic oscillator. Measuring the behavior of the pendulum on a set of example trajectories then allows one to fit the model parameters to get good agreement between model and observed behavior. Since our analog neuromophic core attempts to replicate the dynamics of certain idealized neuron models, we are in a similar situation. Part of the correspondence between our physical substrate and the model is ensured by *calibration*. During task specific training, we adapt a model to a specific hardware instance, by training *in the loop*.

The *in-the-loop* training paradigm relies on the fact, as we alluded to above, that it is possible to use the parameter gradient computed based on measurements and a model of a physical system to update the parameters of the physical system in a composable fashion. This is the basis of all three gradient-based learning paradigms realized in the BrainScaleS-2 system so far, namely

- Time-to-first spike (Göltz et al., 2021)
- Surrogate-Gradient-Based Learning (Cramer et al., 2022)
- Analog ANN training (Weis et al., 2020)

They differ in which measurements are necessary and what model of the physical system is used. In the time-to-first spike gradient-based training scheme, which we won't discuss in detail here, the essential idea is that it is possible to compute the derivative of the spike time with respect to input weights based on an analytical expression of the spike time. In other words the only measurements required of the system are the spike times of the neurons present in the (feed-forward) network. The model assumes that the physical system evolves according to dynamics with certain ratios between synaptic and membrane time-constants, which need to be ensured by calibration. In the surrogate gradient paradigm the network dynamics is modeled by a recurrent neural network (RNN) closely corresponding to the continuous time dynamic. It requires the observation of the

membrane voltages of all neurons of the network with a temporal resolution comparable to that of the chosen RNN timestep. We will discuss this approach in more detail in Section 3.3.1. Finally in the analog ANN training mode the behavior of the system is modeled by a linear operation (implemented by the synaptic crossbar), together with a non-linearity (implemented by the digital processor and the analog readout). Again the actual correspondence of the system to this behavior needs to be ensured by calibration. In order to estimate gradients the layerwise results are needed and a full precision version of the implemented operation is used to propagate errors between layers. We give a more detailed description of this approach in Section 3.3.2.

3.3.1. Surrogate-Gradient-Based Training of SNNs

Gradient-based training of SNNs has historically been impeded by their binary nature, and was mostly limited to rate-based codings schemes. Surrogate-gradient-based approaches have only recently enabled the optimization of SNNs eliciting temporally sparse spiking activity (Neftci et al., 2019). To that end, these approaches attach a modified derivative to the neurons' activation functions and thereby smoothen their gradients. Relying on the temporally resolved membrane potential, these surrogate gradients can often be easily computed for numerically evolved SNNs. For analog systems, the neurons' membrane potentials evolve as physical quantities and are hence not directly available for the respective gradient computation; digitization is complicated by the intrinsic parallelism of such devices. BrainScaleS-2, however, does allow for the parallel digitization of membrane traces, despite its accelerated nature. For this purpose, we employed the massively parallel ADCs and scheduled their conversion via the on-chip PPUs. This allowed us to parallelly digitize the temporal evolution of the membrane potentials of 256 neurons with a sampling period of 1.7 µs. Based on the digitized membrane traces and spike times, we then constructed a PyTorch computation graph based on the LIF equations. By incorporating the actual, measured traces into this model of our system, we aligned the computation graph with the actual dynamics of our silicon neurons. Our framework, hence, effectively attached gradients to the otherwise non-differentiable physical dynamics and allowed to minimize arbitrary loss functions via backpropagation through time (BPTT) in combination with state-of-the-art optimizers (Kingma and Ba, 2015).

We benchmarked our learning framework on several challenging datasets. For example, we trained feed-forward SNN with a hidden layer composed of 246 LIF neurons on the handwritten MNIST digits (LeCun et al. 1998, **Figure 8A**), where we reached a test accuracy of (97.6 ± 0.1) % (Cramer et al., 2022). Notably, we observed a time-to-decision of less than 7 µs (**Figure 8B**). We exploited this low classification latency in a fast inference mode, where we artificially reset the analog neuronal states to prepare the network for the subsequent input sample, and reached a classification throughput of 84 k images per second (**Figure 8C**). The flexibility of our framework could furthermore be demonstrated by augmenting the loss term with a sparsity penalty. This regularization allowed us to perform inference on the MNIST data with on average only 12 hidden layer spikes per image (**Figure 8D**). Moreover, our framework could be extended to facilitate the training of recurrent SNNs. Specifically, we trained a recurrent SNN with a single hidden layer composed of 186 LIF neurons on the SHD dataset (Cramer et al., 2020b), where we reached a test performance of (80.0 ± 1.0) % (Cramer et al., 2022).

3.3.2. Artificial Neural Networks on BrainScaleS-2

Extending its application into the realm of non-spiking neural networks, BrainScaleS-2 also allows processing artificial neural networks within its analog core. This yields several advantages, such as the possibility to process large amounts of input data using convolutional neural networks, and easier multiplexing of the available resources due to the non-time-continuous fashion of the underlying multiply-accumulate operation. On BrainScaleS-2, input vectors are encoded as arrays of graded spikes, which control the activation time of synapses. The versatility of the neuron circuits allows them to act as integrators without temporal dynamics, simply accumulating synaptic currents (cf. Section 2.2 and **Figure 9**). The voltage on the membrane capacitances is finally digitized using the 8 bit columnar ADC. The possibility to configure neurons independently allows for a hybrid operating mode with parts of the chip processing ANN layers and other parts processing a spiking network—a feature unique among accelerators based on analog computation. A detailed description of this operating mode and the corresponding software interfaces is given by Spilger et al. (2020) and Weis et al. (2020).

Our software interface enables training of ANNs on BrainScaleS-2 within the PyTorch framework (Paszke et al., 2019) and thereby benefits from well-established gradient-based training methodologies. We run the forward path of the network on hardware and calculate weight updates on a host computer, assuming an ideal linear model for all computational elements. The chip is supplied with appropriately-sized MAC operations and executes those in the analog core, the PPU handles simple operations on the results, like pooling or applying activation functions. Experiment control, like splitting ANNs into simple MAC operations, is usually handled on the host computer.

As an initial proof-of-concept for analog matrix-vector multiplication on BrainScaleS-2, we showcase a simple classifier for the MNIST dataset of handwritten digits (LeCun et al., 1998). Using a three-layer convolutional model, we achieve an accuracy of 98.0 % after training with hardware in-the-loop (Weis et al., 2020). The same model reaches 98.1 % accuracy on a CPU when discretized to the same 6 bit weight resolution. This operating mode was further used to classify the human activity recognition dataset (Spilger et al., 2020) and to detect atrial fibrillation in electrocardiogram traces in Stradmann et al. (2021).

In summary, we have shown that the analog network core of the BrainScaleS-2 system—in addition to the prevailing spiking operation—can successfully perform vector-matrix multiplications. Applying this feature to classical ANNs, competitive classification precision has been reached. While the current proof-of-concept implementation of this operating mode still carries large potential for future optimizations, interesting

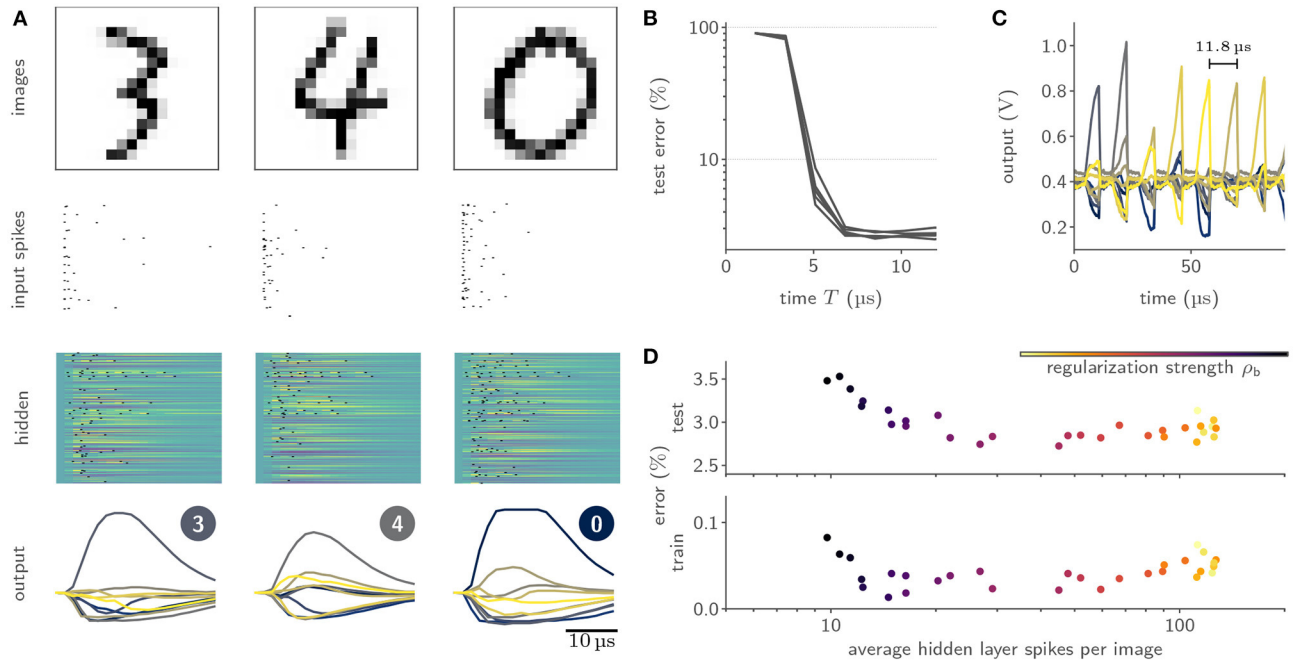

FIGURE 8 | Training of and inference with SNNs on BrainScaleS-2 using surrogate gradients. **(A)** Exemplary activity snapshots for three 16 × 16 pixel MNIST images, the resulting spike-latency encodings, an overlay of the hidden layer membrane traces recorded at a data rate of 1.2 Gbit/s as well as the resulting hidden layer spikes, and the output layer membrane traces. **(B)** Spike-latency encoding promotes fast decision times: after 7 µs the network reaches its peak performance. **(C)** A fast inference mode allows to exploit the quick decision times by artificially resetting the neuronal states after 8 µs and therefore preparing the network for presentation of the next sample, culminating in a classification rate of 85 k inferences per second. **(D)** The PyTorch-based framework allows to co-optimize for near-arbitrary regularization terms, including sparsity penalties. In this instance, BrainScaleS-2 can be trained to classify MNIST images with an average of 12 hidden layer spikes per image without a significant decline in performance.

hybrid applications combining spiking and non-spiking network layers are already possible with the current hardware generation.

4. A PRINCIPLED APPROACH TO GRADIENT-BASED PARAMETER OPTIMIZATION IN NEUROMORPHIC SYSTEMS

Given the multitude of approaches to learning and parameter optimization in use in the neuromorphic computing and computational neuroscience community, a natural question arises: Is there a principled way to understand at least gradient-based optimization in parametrized physical systems and (as a particular case) neuromorphic hardware. Here we want to argue that there is such an approach and that it is particularly useful for neuromorphic hardware with complex neuron dynamics and plasticity. As already discussed in the preceding Section 3.3, the key to estimating gradients in a physical system is an appropriate choice of model. The first observation is that most neuromorphic hardware, and in particular the BrainScaleS-2 system, is well described as a hybrid dynamical system.[1] That is, their dynamics are described by differential equations—the neuron equations, the equations for the correlation traces—together with state transitions—spike-based synaptic input, neuron reset—which happen when jump conditions—a membrane voltage crosses its threshold—are satisfied. The task of estimating gradients in neuromorphic hardware is therefore mainly subsumed under the question of how to compute parameter gradients in a hybrid dynamical system, which is a well-established subject (De Backer, 1964; Rozenvasser, 1967; Galn et al., 1999; Barton and Lee, 2002). The second observation is that the dynamics in spiking neural networks decouple, except for spike times, and most of the parameters (the synaptic weights) only enter the state transition functions. Moreover, the jump conditions typically only depend on the state variables of single neuron circuits. These two facts taken together result in simple event-based rules for gradient computation.

More formally a hybrid dynamical system is given by differentiable functions $f^{(s)}(x, p, t)$, labeled by the state s the system is in, which specify the dynamics of the state vector x, while in this state:

$$\dot{x} = f^{(s)}(x, p, t) \tag{4}$$

together with jump conditions $j_r^{(s)}(x, p, t) = 0$ and transition equations $x^+ = T_r^{(s)}(x^-, p, t)$. We use p to indicate a dependence on some number of parameters of both the dynamics, jump condition and transition equations. One simple example would

[1]In the case of fully digital neuromorphic hardware, the situation is more complicated, as they implement a numerical solver for a hybrid dynamical system.

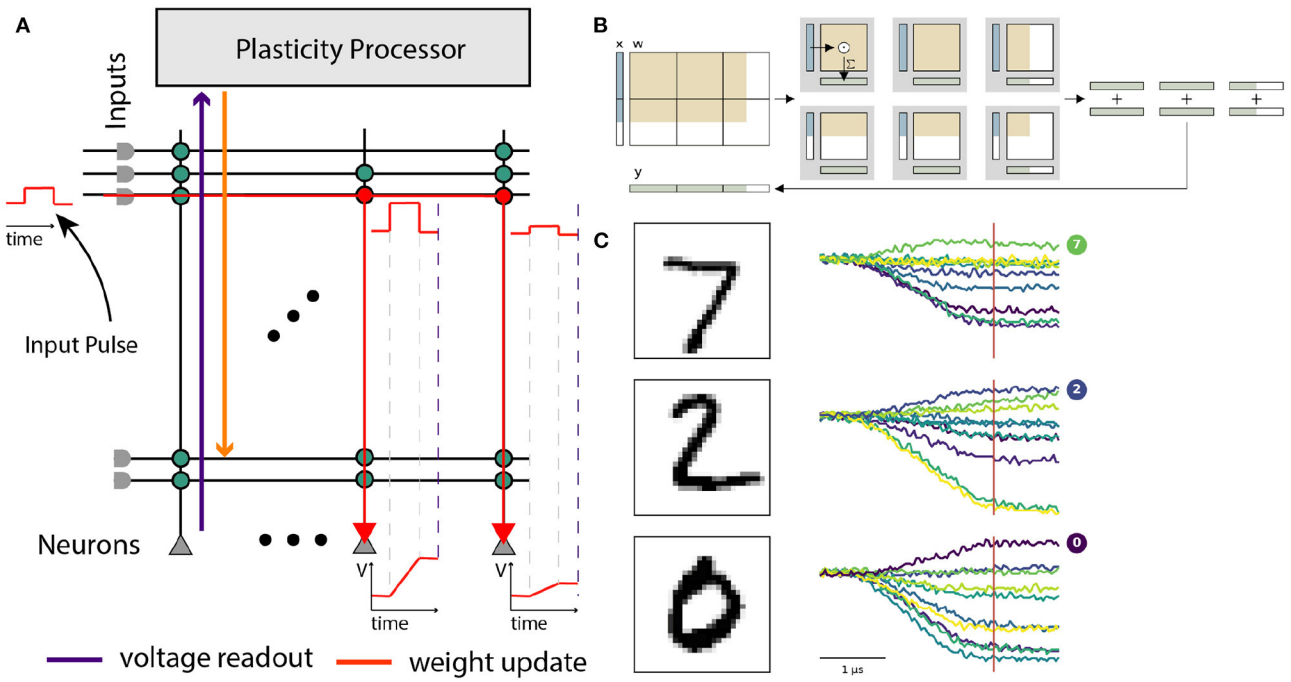

FIGURE 9 | Use of the BrainScaleS-2 system in ANN operating mode. **(A)** Analog input values are represented as pulses sent consecutively into the synaptic crossbar. The duration of these pulses represents input activations. Synapses modulate the pulse height depending on the stored weight. Signed weights can be achieved by using two synapse rows, excitatory and inhibitory, for the same input. The neuron circuits serve as an integrator of the generated currents. Readout of the voltage occurs in parallel once triggered (dashed violet line) by the columnar ADC. **(B)** Layers that exceed the hardware resources are tiled into hardware execution instances and sequentially executed. Panel taken from Spilger et al. (2020). **(C)** Example membrane traces during inference of MNIST handwritten digits in the output layer. The ten output activations are sampled at the indicated time for all neurons in parallel.

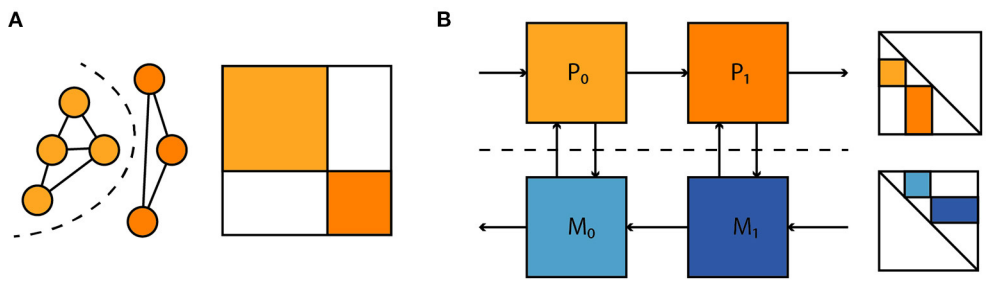

FIGURE 10 | (A) A large dynamical system consisting of two decoupled subsystems. It has a block diagonal Jacobian and corresponding decoupled sensitivity equations. **(B)** Parameter gradient computation in sequentially composed physical systems (orange) can be performed by composing gradient computation in models of the physical systems (blue).

be the Leaky Integrate and Fire neuron model. The state is given by membrane voltage and synaptic input current $x = (V, I)$ of N neurons,

$$\tau_m \dot{V} = (V_L - V) + RI \tag{5}$$

$$\tau_s \dot{I} = -I \tag{6}$$

Each jump condition corresponds to the membrane threshold crossing condition of one of the N neurons

$$V_i^- - (V_T)_i = 0 \tag{7}$$

and the corresponding transition equation implements the reset of membrane voltage and the jump of the synaptic input current, when neuron i fires:

$$I^+ = I^- + We_i \tag{8}$$

$$V_i^+ = V_{\text{reset}}, \tag{9}$$

here W denotes the synaptic weight matrix and e_i is the i-th unit vector. The non-zero entries of the Jacobian

$$J^{(s)} = \partial_x f^{(s)}(x, p, t) \tag{10}$$

characterizes which dynamical variables x_i directly couple to each other. A distinguishing feature of all neuromorphic architectures is that even though the overall state space might be large (1×10^5 to 1×10^6 of neuron equations in the case of large scale systems like Loihi, TrueNorth, SpiNNaker and the WaferScale BrainScaleS system), the Jacobian $J^{(s)}$ is sparse and block diagonal. Indeed a system of N Leaky Integrate and Fire neurons has a Jacobian J with diagonal entries

$$\begin{pmatrix} -1/\tau_m & R/\tau_m \\ 0 & -1/\tau_s \end{pmatrix} \tag{11}$$

The Jacobian of a jump condition

$$\partial_x j^{(s)}(x, p, t) \tag{12}$$

in the case of spiking neural networks is equally sparse, for a Leaky Integrate and Fire neuron model with threshold V_T in order for the k-th neuron to spike

$$j^k(x, p, t) = V_k - V_T \tag{13}$$

and therefore has only one non-zero entry.

A key observation is that if the state of the system is known at a time t_0, then the time t^* at which a transition happens is an implicit function of the system's state in a neighborhood close to the transition. One can therefore use the *implicit function* theorem (under certain technical conditions), to compute the parameter derivative $\frac{\mathrm{dt}^*}{\mathrm{dp}}$:

$$\partial_x j \left[\left(\frac{\partial x}{\partial p} + \partial_t x \right) + \partial_p j + \frac{\mathrm{dt}^*}{\mathrm{dp}} \right] = 0 \tag{14}$$

In contrast to the time-to-first spike approach (Göltz et al., 2021), this does not require explicit or analytical knowledge of the function $t^\star(x, p)$ and is also applicable to more complex neuron models. In the context of spiking neural networks, this was recognized by Wunderlich and Pehle (2020) and elaborated in full generality by Pehle (2021). Concurrent work also introduced this technique to the wider machine learning community (Chen et al., 2021). This observation is particularly useful in the case of spiking neurons because the jacobian $\partial_x j$ is sparse and therefore results in a sparse coupling of the gradient computation across jumps.

To solve computational tasks, the parameters p of a spiking neural network, such as the weight matrix W, need to be optimized according to some (differentiable) loss function. Given such a particular model and task, a constrained optimization problem can be formulated for the parameters p, involving an integral over a task-specific loss function l

$$L = \int_0^T l(x, p, t)\mathrm{dt}, \tag{15}$$

subject to the constraints on the state x given by the equations above. The calculation of the gradient of the loss with respect to the parameters involves the *adjoint equations*

$$\lambda'^T = \lambda^T J + \partial_x l, \tag{16}$$

where the jacobian J of the dynamical system ensures that the computation has the same sparse coupling pattern as the forward equations and $()'$ indicates the derivative *reverse* in time. By their nature, typical neuromorphic architectures and BrainScaleS-2, in particular, have $O(n)$ parameters, where n is the number of "neuron" circuits, which enter the continuous-time evolution. More specifically, in the case of BrainScaleS-2, those are the calibration parameters of the neuron circuits as discussed in Section 2.2 or rather the model parameters (depending on the viewpoint). A much larger fraction of the parameters $O(n^2)$, namely the synaptic weights, enter only the transition equations.

A similar argument to the one made above for the parameter derivative of the transition times allows one to then relate the adjoint state variables after λ^+ to the adjoint state variables before the transition λ^- and yields an *event-based* rule for gradient accumulation of the parameters that only enter the transition equations (in particular the synaptic weights). This is elaborated more explicitly in Wunderlich and Pehle (2020) and Pehle (2021). The event-based nature of the gradient accumulation and the sparse propagation of error information has immediate consequences for neuromorphic hardware. In particular, it means that only *sparse observations* or measurements are necessary to estimate the gradients successfully, which is a significant advantage over the surrogate gradient approach of Section 3.3.1, which (at least currently) requires dense observations of membrane voltages.

In the context of the *in-the-loop* training paradigm, the general framework sketched here also has attractive consequences. As the numerical implementation and the implemented dynamical system are separate, one can choose appropriate integration methods, such as ones also applicable to multi-compartment neuron models. There is a well-understood way in which numerical implementation of the forward and reverse time dynamics are related (Haier et al., 2006 II.3, note that "adjoint" there is not used quite in the same sense as here). Since arbitrary loss functions are supported, the hybrid dynamical system used to model the neuromorphic substrate can receive both model and task-specific loss contributions. In particular arbitrary temporally sparse and partial system observations, $\hat{x}_i$ at times t_i can enter a loss term of the form

$$l_m = \sum_i |Px - P\hat{x}_i|^2 \delta(t - t_i), \tag{17}$$

where P denotes a linear projection to a state subspace. Such loss terms allow one to account for parameter mismatch and dynamical differences between model and hardware without a need for full access to the system state, which is prohibitive for large neuromorphic systems. Similarly, observations of spikes alone could be used to both fit the model to the neuromorphic substrate and enter the computation of parameter gradients without additional other observations. Moreover, as illustrated in **Figure 10** this extends to situations where one has separate models for physical subsystems—dynamical models, and adjoint computation of parameter gradients compose as one would expect. This compositionality is useful for the time-multiplexed execution of large feedforward

models and the coupling of different parametrized physical substrates, with the goal of task-specific end-to-end optimization. In summary we believe this to be a promising approach to gradient-based optimization in neuromorphic hardware and intend to apply it to the BrainScaleS-2 architecture presented here.

5. DISCUSSION

We have presented the BrainScaleS-2 system architecture as implemented in a single-chip ASIC with an analog core consisting of 512 neurons and 2^{17} synapses, as well as two embedded plasticity and control processors. The system design meets our expectations concerning flexibility and configurability of the analog components and has proven to be a versatile platform for implementing a wide array of tasks across several domains.

The modularity of the architecture, in particular, the neuron circuits, allows for the evaluation of the implemented neuron models on a wide range of parameters as seen in Section 3.1 and even accelerated emulation of multi-compartment neurons (Kaiser et al., 2021). The hybrid approach of combining the analog core with flexible digital control and plasticity architecture has enabled many of the experiments reported here. Beyond the immediate practical benefits, it also lays the foundation toward a fully integrated standalone deployment of this neuromorphic architecture, as partially realized by the mobile system (Stradmann et al., 2021). It is also crucial for the scalable calibration and control of a larger scale architecture, which would use the presented neuromorphic core as a unit of scale.

Another aspect of implementing a large-scale system based on the presented architecture has not been touched upon: implementing a scalable event-routing architecture and wafer-scale integration. Since the spiking network is emulated completely asynchronously, it has punishing demands on latency and timing jitter. Future work will have to address this challenge head-on. While scaling to a larger reticle size will be straightforward, accomplishing wafer-scale integration and networking between wafers requires further work and funding. However, we believe that our experience with the first-generation BrainScaleS wafer-scale system puts us in an excellent position to accomplish a wafer-scale second-generation design (Schemmel et al., 2010). As an intermediate step, we plan to realize a multi-chip system building upon existing BrainScaleS-1 wafer-scale system components. Interconnectivity will be implemented using FPGAs based on the EXTOLL network protocol (Resch et al., 2014; Neuwirth et al., 2015). Concurrently we are working on a low-power inter-chip link, which eventually could also be the basis for on-wafer connectivity. While wafer-scale integration is surely a valuable step to a scale-up of the system, integrating multiple wafer-scale systems using conventional communication technology will present additional challenges. Furthermore, assembling wafer-scale systems is a time-consuming and difficult task. We explored techniques such as embedding wafers into printed circuit boards to simplify this task (Zoschke et al., 2017).

Several digital neuromorphic architectures have been proposed in recent years (Furber et al., 2012; Merolla et al., 2014; Davies et al., 2018; Frenkel et al., 2018, 2019; Mayr et al., 2019; Pei et al., 2019). Instead of analog emulation of the neuron and synapse dynamics, these commonly rely on a digital implementation with varying biological faithfulness and flexibility. Other architectures incorporate—similar to the BrainScaleS platform—an analog emulation of neuronal and synaptic dynamics (Benjamin et al., 2014; Moradi and Indiveri, 2014; Moradi et al., 2018; Neckar et al., 2018; Rahimi Azghadi et al., 2020); a review of earlier approaches were assembled by Indiveri et al. (2011). Some of them are directly inspired by the pioneering work of Mead and Mahowald (1988) and Mead (1990) and rely on subthreshold characteristics of transistors. Digital and analog verification methods for the BrainScales-2 system were previously discussed in Grübl et al. (2020). A description of the BrainScaleS-2 architecture, event routing, block diagrams, analog-matrix multiplication extensions, on-hardware measurements of matrix multiplication, and the multi-compartment extension, as well as a first application of the artificial neural network operation mode has been given in Schemmel et al. (2020).

Considerable progress on learning methods for neuromorphic hardware has been made as well. Here we only highlight those most closely related to the presented methods. Esser et al. (2016) proposed the use of a pseudo-derivative to replace the derivative of the Heaviside function used for spike threshold detection and also used a forward pass simulating the precision constraints (trinary synapses) of the target hardware and floating-point precision backward pass. Subsequent work applied this training procedure to other neuron models or used other pseudo-derivatives (Bellec et al., 2018; Neftci et al., 2019). The in-the-loop training approach was realized on the BrainScaleS platform by Schmitt et al. (2017) for rate-based models and (Cramer et al., 2022) for SNNs employing individual spikes for information transfer and processing, as summarized in Section 3.3.1.

Petrovici et al. (2016) related stochastically stimulated and recurrently connected populations of LIF neurons in the high-conductance state to Restricted Boltzmann Machines. This relationship suggests finding variational representations of and to sample from high dimensional probability distributions on BrainScaleS-2. The accelerated nature of the system allows one to rapidly produce samples over long periods. Recently such a variational representation was used to represent POVM probability distributions of states in certain quantum systems on BrainScaleS-2 (Czischek et al., 2019; Klassert et al., 2021; Czischek et al., 2022).

Beyond the experiments presented here, we believe the architecture can serve as a versatile platform for further experiments in multiple directions. One promising direction appears to be to close the loop on two different levels of the system hierarchy, as has been explored as part of the learning-to-learn experiments (Bohnstingl et al., 2019). Similarly, the evolving-to-learn framework by Jordan et al. (2020) seems predestined for implementation on BrainScaleS-2. The accelerated nature of the system makes it ideal for rapid execution of an inner-loop, involving a bio-inspired

learning rule and an additional optimization technique in the outer loop. The flexible architecture for plasticity processing allows one to for example evaluate RNN-based plasticity rules. In another direction online learning rules, such as E-Prop (Bellec et al., 2020) are natural candidates for implementation on BrainScaleS-2. We have found a way to adopt such learning algorithms to our platform, taking into account the constraints on the observability of correlation and neuron membrane traces.

By providing this platform, we hope to establish physical modeling as a versatile tool to the machine learning and computational neuroscience community. Furthermore, it can not yet be foreseen what deep insights into the collective dynamics of neural networks will be revealed by emulating multiple, interconnected layers of structured neurons, all continuously subjected to complex learning rules and homeostatic processes.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. This data can be found here: http://yann.lecun.com/exdb/mnist/; https://zenkelab.org/resources/spiking-heidelberg-datasets-shd/.

AUTHOR CONTRIBUTIONS

CP drafted the article and authored all sections except Accelerated Analog Emulation of Neural Dynamics, Faithful Emulation of Complex Neuron Dynamics, Collective Dynamics, Surrogate-Gradient-Based Training of SNNs, Artificial Neural Networks on BrainScaleS-2, contributed to the digital design and verification of the BrainScaleS-2 ASIC, conceived and implemented the processor memory interface that enable the experiments reported in sections Surrogate-Gradient-Based Training of SNNs, Artificial Neural Networks on BrainScaleS-2, implemented the FPGA I/O interface and Spike router that enabled the experiment reported in section Accelerated Closed Loop Robotics, and conceptualized the approach described in section A Principled Approach to Gradient-Based Parameter Optimization in Neuromorphic Systems. SB conceived and implemented parts of the analog neuromorphic core, most notably the analog neuron circuits, evaluated their performance on the AdEx firing patterns and contributed to the software stack. SB and BC conceptualized, implemented, and evaluated the surrogate-gradient-based learning framework. BC conceived, implemented, and evaluated the experiments on reservoir computing. KS conceptualized, implemented, evaluated all experimental aspects of the accelerated insect navigation and robotics experiments, conceptualized, characterized and implemented the CADC full-custom circuits with the exception of an operational amplifier by HBP collaboration partners from the Sabancı Üniversitesi in Istanbul and another operational amplifier from Hock (2014), designed and implemented both the chip-carrier and interface-boards for the BrainScaleS-2 ASIC, contributed additional work to the analog design and tape-out of the BrainScaleS-2 ASIC. YS contributed to the design, commissioning of the BrainScaleS-2 ASIC and the associated software stack. JW developed calibration routines, commissioned the non-spiking operating mode, conducted experiments using ANNs and contributed to the software stack. JK implemented and evaluated the experiments on multi-compartment neurons and contributed to the software stack. AL ported the insect navigation task to the full-scale system, developed on-chip calibration routines for neuron circuits and contributed to the software stack. EM provided conceptual and scientific advice and is the lead developer and architect of the BrainScaleS-2 software stack. JS wrote the initial extended abstract for the article and is the lead designer and architect of the BrainScaleS-2 neuromorphic system. All authors contributed to and edited the final manuscript.

FUNDING

The research has received funding from the EC Horizon 2020 Framework Programme under Grant Agreements 720270, 785907, and 945539 (HBP), the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy EXC 2181/1-390900948 (the Heidelberg STRUCTURES Excellence Cluster), the German Federal Ministry of Education and Research under grant number 16ES1127 as part of the Pilotinnovationswettbewerb Energieeffizientes KI-System from the Helmholtz Association's Initiative and Networking Fund under project number SO-092 (Advanced Computing Architectures, ACA), the Manfred Stärk Foundation and the Lautenschläger-Forschungspreis 2018 for Karlheinz Meier.

ACKNOWLEDGMENTS

The authors wish to thank all present and former members of the Electronic Vision(s) research group contributing to the BSS-2 hardware and software system. Special thanks to Syed Aamir, Simon Friedmann, Andreas Hartel, Vitali Karasenko, Gerd Kiene, Matthias Hock, and Andreas Gróbl for contributing to the digital and analog hardware design, testing and initial software development, Christian Mauch, Oliver Breitwieser, Arne Emmel, and Philipp Spilger for the production software stack and system operation. We would like to thank Friedemann Zenke and Mihai Petrovici for conceptual advice and inspiration. We thank S. Höppner and S. Scholze from the group Hochparallele VLSI-Systeme und Neuromikroelektronik of C. Mayr from TU Dresden for the PLL macro cell (Höppner et al., 2013) and SerDes macros (Scholze et al., 2012) used in the current BSS-2 chip revisions, and Tugba Demirci from EPFL Lausanne for the on-chip fast MADC.

REFERENCES

Aamir, S. A., Müller, P., Kiene, G., Kriener, L., Stradmann, Y., Grübl, A., et al. (2018a). A mixed-signal structured AdEx neuron for accelerated neuromorphic cores. *IEEE Trans. Biomed. Circ. Syst*. 12, 1027–1037. doi: 10.1109/TBCAS.2018.2848203

Aamir, S. A., Stradmann, Y., Müller, P., Pehle, C., Hartel, A., Grübl, A., et al. (2018b). An accelerated LIF neuronal network array for a large-scale mixed-signal neuromorphic architecture. *IEEE Trans. Circ. Syst*. 65, 4299–4312. doi: 10.1109/TCSI.2018.2840718

Barnett, L., Lizier, J. T., Harré, M., Seth, A. K., and Bossomaier, T. (2013). Information flow in a kinetic ising model peaks in the disordered phase. *Phys. Rev. Lett*. 111, 177203. doi: 10.1103/PhysRevLett.111.177203

Barton, P. I., and Lee, C. K. (2002). Modeling, simulation, sensitivity analysis, and optimization of hybrid systems. *ACM Trans. Model. Comput. Simul*. 12, 256–289. doi: 10.1145/643120.643122

Bellec, G., Salaj, D., Subramoney, A., Legenstein, R., and Maass, W. (2018). Long short-term memory and learning-to-learn in networks of spiking neurons. *arXiv* preprint arXiv:1803.09574.

Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., et al. (2020). A solution to the learning dilemma for recurrent networks of spiking neurons. *Nat. Commun*. 11, 1–15. doi: 10.1038/s41467-020-17236-y

Benjamin, B. V., Gao, P., McQuinn, E., Choudhary, S., Chandrasekaran, A. R., Bussat, J.-M., et al. (2014). Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proc. IEEE* 102, 699–716. doi: 10.1109/JPROC.2014.2313565

Bi, G. Q., and Poo, M. M. (1998). Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *J. Neurosci*. 18, 10464–10472. doi: 10.1523/JNEUROSCI.18-24-10464.1998

Billaudelle, S., Cramer, B., Petrovici, M. A., Schreiber, K., Kappel, D., Schemmel, J., et al. (2021). Structural plasticity on an accelerated analog neuromorphic hardware system. *Neural Netw*. 133, 11–20. doi: 10.1016/j.neunet.2020.09.024

Billaudelle, S., Stradmann, Y., Schreiber, K., Cramer, B., Baumbach, A., Dold, D., et al. (2020). "Versatile emulation of spiking neural networks on an accelerated neuromorphic substrate," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)* (Seville: IEEE).

Bohnstingl, T., Scherr, F., Pehle, C., Meier, K., and Maass, W. (2019). Neuromorphic hardware learns to learn. *Front. Neurosci*. 13, 483. doi: 10.3389/fnins.2019.00483

Brette, R., and Gerstner, W. (2005). Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *J. Neurophysiol*. 94, 3637–3642. doi: 10.1152/jn.00686.2005

Chen, R. T. Q., Amos, B., and Nickel, M. (2021). "Learning neural event functions for ordinary differential equations," in *International Conference on Learning Representations*. Vienna.

Cramer, B., Billaudelle, S., Kanya, S., Leibfried, A., Grübl, A., Karasenko, V., et al. (2022). Surrogate gradients for analog neuromorphic computing. *Proc. Natl. Acad. Sci. U.S.A*. 119. doi: 10.1073/pnas.2109194119

Cramer, B., Stöckel, D., Kreft, M., Wibral, M., Schemmel, J., Meier, K., et al. (2020a). Control of criticality and computation in spiking neuromorphic networks with plasticity. *Nat. Commun*. 11, 2853. doi: 10.1038/s41467-020-16548-3

Cramer, B., Stradmann, Y., Schemmel, J., and Zenke, F. (2020b). "The heidelberg spiking data sets for the systematic evaluation of spiking neural networks," in *IEEE Transactions on Neural Networks and Learning Systems*.

Czischek, S., Baumbach, A., Billaudelle, S., Cramer, B., Kades, L., Pawlowski, J. M. et al., (2022). Spiking neuromorphic chip learns entangled quantum states. *SciPost Physics*. 12, 039.

Czischek, S., Pawlowski, J. M., Gasenzer, T., and Gärttner, M. (2019). Sampling scheme for neuromorphic simulation of entangled quantum systems. *Phys. Rev. B* 100, 195120. doi: 10.1103/PhysRevB.100.195120

Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., Choday, S. H., et al. (2018). Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro* 38, 82–99. doi: 10.1109/MM.2018.112130359

De Backer, W. (1964). Jump conditions for sensitivity coefficients. *IFAC Proc*. 1, 168–175. doi: 10.1016/S1474-6670(17)69603-4

Esser, S. K., Merolla, P. A., Arthur, J. V., Cassidy, A. S., Appuswamy, R., Andreopoulos, A., et al. (2016). Convolutional networks for fast, energy-efficient neuromorphic computing. *Proc. Natl. Acad. Sci. U.S.A*. 113, 11441–11446. doi: 10.1073/pnas.1604850113

Frenkel, C., Lefebvre, M., Legat, J.-D., and Bol, D. (2018). A 0.086-mm^{2}12.7-pj/sop 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm cmos. *IEEE Trans. Biomed. Circ. Syst*. 13, 145–158. doi: 10.1109/TBCAS.2018.2880425

Frenkel, C., Legat, J.-D., and Bol, D. (2019). Morphic: A 65-nm 738k-synapse/mm^2 quad-core binary-weight digital neuromorphic processor with stochastic spike-driven online learning. *IEEE Trans. Biomed. Circ. Syst*. 13, 999–1010. doi: 10.1109/TBCAS.2019.2928793

Friedmann, S. (2013). *A New Approach to Learning in Neuromorphic Hardware* (Ph.D. thesis). Ruprecht-Karls-Universität Heidelberg.

Friedmann, S., Schemmel, J., Grübl, A., Hartel, A., Hock, M., and Meier, K. (2017). Demonstrating hybrid learning in a flexible neuromorphic hardware system. *IEEE Trans. Biomed. Circ. Syst*. 11, 128–142. doi: 10.1109/TBCAS.2016.2579164

Furber, S. B., Lester, D. R., Plana, L. A., Garside, J. D., Painkras, E., Temple, S., et al. (2012). Overview of the SpiNNaker system architecture. *IEEE Trans. Comput*. 99, 2454–2467. doi: 10.1109/TC.2012.142

Galn, S., Feehery, W. F., and Barton, P. I. (1999). Parametric sensitivity functions for hybrid discrete/continuous systems. *Appl. Numer. Math*. 31, 17–47. doi: 10.1016/S0168-9274(98)00125-1

Göltz, J., Kriener, L., Baumbach, A., Billaudelle, S., Breitwieser, O., Cramer, B., et al. (2021). Fast and energy-efficient neuromorphic deep learning with first-spike times. *Nat. Mach. Intell*. 3, 823–835. doi: 10.1038/s42256-021-00388-x

Grübl, A., Billaudelle, S., Cramer, B., Karasenko, V., and Schemmel, J. (2020). Verification and design methods for the brainscales neuromorphic hardware system. *J. Signal Process. Syst*. 92, 1277–1292. doi: 10.1007/s11265-020-01558-7

Haier, E., Lubich, C., and Wanner, G. (2006). *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Heidelberg: Springer.

Harris, T. E. (2002). *The Theory of Branching Processes*. Courier Corporation. Berlin, Göttingen, Heidelberg: Springer Verlag OHG. Available online at: https://www.rand.org/content/dam/rand/pubs/reports/2009/R381.pdf

Hock, M. (2014). *Modern Semiconductor Technologies for Neuromorphic Hardware* (Ph.D. thesis). Ruprecht-Karls-Universität Heidelberg.

Hock, M., Hartel, A., Schemmel, J., and Meier, K. (2013). "An analog dynamic memory array for neuromorphic hardware," in *Circuit Theory and Design (ECCTD), 2013 European Conference on*. Dresden 1–4.

Höppner, S., Eisenreich, H., Henker, S., Walter, D., Ellguth, G., and Schüffny, R. (2013). A compact clock generator for heterogeneous gals mpsocs in 65-nm cmos technology. *IEEE Trans. Very Large Scale Integr. Syst*. 21, 566–570. doi: 10.1109/TVLSI.2012.2187224

Indiveri, G., Linares-Barranco, B., Hamilton, T. J., van Schaik, A., Etienne-Cummings, R., Delbruck, T., et al. (2011). Neuromorphic silicon neuron circuits. *Front. Neurosci*. 5:73. doi: 10.3389/fnins.2011.00073

Jaeger, H. (2001). *The "echo state" approach to analysing and training recurrent neural networks*. Technical Report GMD Report 148, German National Research Center for Information Technology.

Jordan, J., Schmidt, M., Senn, W., and Petrovici, M. A. (2020). Evolving to learn: discovering interpretable plasticity rules for spiking networks. *arXiv preprint* arXiv:2005.14149. doi: 10.7554/eLife.66273.sa2

Kaiser, J., Billaudelle, S., Müller, E., Tetzlaff, C., Schemmel, J., and Schmitt, S. (2021). *Emulating Dendritic Computing Paradigms on Analog Neuromorphic Hardware*. Publication in Neuroscience. doi: 10.1016/j.neuroscience.2021.08.013 Available online at: https://www.sciencedirect.com/science/article/pii/S0306452221004218

Karasenko, V. (2020). *Von Neumann bottlenecks in non-von Neumann computing architectures* (Ph.D. thesis). Heidelberg University.

Kingma, D. P., and Ba, J. (2015). *3rd International Conference for Learning Representations*, San Diego.

Klassert, R., Baumbach, A., Petrovici, M. A., and Gärtner, M. (2021). *Variational Learning of Quantum Ground States on Spiking Neuromorphic Hardware*. Available online at: https://ssrn.com/abstract=4012184

Larkum, M. E., Zhu, J. J., and Sakmann, B. (1999). A new cellular mechanism for coupling inputs arriving at different cortical layers. *Nature* 398, 338–341. doi: 10.1038/18686

LeCun, Y., Cortes, C., and Burges, C. J. C. (1998). The MNIST database of handwritten digits. New York, USA. Available online at: http://yann.lecun.com/exdb/mnist/

Leibfried, A. (2021). *On-chip calibration and closed-loop experiments on analog neuromorphic hardware* (Master's thesis). Heidelberg University.

Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural Comput*. 14, 2531–2560. doi: 10.1162/089976602760407955

Major, G., Larkum, M. E., and Schiller, J. (2013). Active properties of neocortical pyramidal neuron dendrites. *Annu. Rev. Neurosci*. 36, 1–24. doi: 10.1146/annurev-neuro-062111-150343

Mayr, C., Hoeppner, S., and Furber, S. (2019). Spinnaker 2: A 10 million core processor system for brain simulation and machine learning. *arXiv preprint* arXiv:1911.02385.

Mead, C. A. (1990). Neuromorphic electronic systems. *Proc. IEEE* 78, 1629–1636. doi: 10.1109/5.58356

Mead, C. A., and Mahowald, M. A. (1988). A silicon model of early visual processing. *Neural Netw*. 1, 91–97. doi: 10.1016/0893-6080(88)90024-X

Merolla, P. A., Arthur, J. V., Alvarez-Icaza, R., Cassidy, A. S., Sawada, J., Akopyan, F., et al. (2014). A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science* 345, 668–673. doi: 10.1126/science.1254642

Moradi, S., and Indiveri, G. (2014). An event-based neural network architecture with an asynchronous programmable synaptic memory. *IEEE Trans. Biomed. Circ. Syst*. 8, 98–107. doi: 10.1109/TBCAS.2013.2255873

Moradi, S., Qiao, N., Stefanini, F., and Indiveri, G. (2018). A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (DYNAPs). *IEEE Trans. Biomed. Circ. Syst*. 12, 106–122. doi: 10.1109/TBCAS.2017.2759700

Müller, E., Mauch, C., Spilger, P., Breitwieser, O. J., Klähn, J., Stöckel, D., et al. (2020). Extending brainscales OS for BrainScaleS-2. *arXiv preprint*.

Munoz, M. A. (2018). Colloquium: Criticality and dynamical scaling in living systems. *Rev. Mod. Phys*. 90, 031001. doi: 10.1103/RevModPhys.90.031001

Naud, R., Marcille, N., Clopath, C., and Gerstner, W. (2008). Firing patterns in the adaptive exponential integrate-and-fire model. *Biol. Cybern*. 99, 335–347. doi: 10.1007/s00422-008-0264-7

Neckar, A., Fok, S., Benjamin, B. V., Stewart, T. C., Oza, N. N., Voelker, A. R., et al. (2018). Braindrop: A mixed-signal neuromorphic architecture with a dynamical systems-based programming model. *Proc. IEEE* 107, 144–164. doi: 10.1109/JPROC.2018.2881432

Neftci, E. O., Mostafa, H., and Zenke, F. (2019). Surrogate gradient learning in spiking neural networks: bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Process. Mag*. 36, 51–63. doi: 10.1109/MSP.2019.2931595

Neuwirth, S., Frey, D., Nuessle, M., and Bruening, U. (2015). "Scalable communication architecture for network-attached accelerators," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)* (Burlingame, CA: IEEE), 627–638.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). "Pytorch: an imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, eds H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Vancouver: Curran Associates, Inc.), 8024–8035

Pehle, C.-G. (2021). *Adjoint equations of spiking neural networks* (Ph.D. thesis). Heidelberg University.

Pei, J., Deng, L., Song, S., Zhao, M., Zhang, Y., Wu, S., et al. (2019). Towards artificial general intelligence with hybrid tianjic chip architecture. *Nature* 572, 106–111. doi: 10.1038/s41586-019-1424-8

Petrovici, M. A., Bill, J., Bytschok, I., Schemmel, J., and Meier, K. (2016). Stochastic inference with spiking neurons in the high-conductance state. *Phys. Rev. E* 94, 042312. doi: 10.1103/PhysRevE.94.042312

Polsky, A., Mel, B. W., and Schiller, J. (2004). Computational subunits in thin dendrites of pyramidal cells. *Nat. Neurosci*. 7, 621–627. doi: 10.1038/nn1253

Power,IS. A. (2010). *PowerISA Version 2.06 Revision b*. Specification, Power.org.

Rahimi Azghadi, M., Chen, Y.-C., Eshraghian, J. K., Chen, J., Lin, C.-Y., Amirsoleimani, A., et al. (2020). Complementary metal-oxide semiconductor and memristive hardware for neuromorphic computing. *Adv. Intell. Syst*. 2, 1900189. doi: 10.1002/aisy.201900189

Resch, M. M., Bez, W., Focht, E., Kobayashi, H., and Patel, N. (2014). "Sustained simulation performance 2014," in *Proceedings of the Joint Workshop on Sustained Simulation Performance, University of Stuttgart (HLRS) and Tohoku University, 2014* (Stuttgart and Tohoku: Springer).

Rozenvasser, E. (1967). General sensitivity equations of discontinuous systems. *Automatika i telemekhanika* 3, 52–56.

Schemmel, J., Billaudelle, S., Dauer, P., and Weis, J. (2020). Accelerated analog neuromorphic computing. *arXiv preprint*.

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

Schiller, J., Major, G., Koester, H. J., and Schiller, Y. (2000). NMDA spikes in basal dendrites of cortical pyramidal neurons. *Nature* 404, 285–289. doi: 10.1038/35005094

Schmitt, S., Klähn, J., Bellec, G., Grübl, A., Güttler, M., Hartel, A., et al. (2017). "Classification with deep neural networks on an accelerated analog neuromorphic system," in *Proceedings of the 2017 IEEE International Joint Conference on Neural Networks*. Anchorage.

Scholze, S., Eisenreich, H., Höppner, S., Ellguth, G., Henker, S., Ander, M., et al. (2012). A 32gbit/s communication soc for a waferscale neuromorphic system. *Integr. VLSI J*. 45, 61–75. doi: 10.1016/j.vlsi.2011.05.003

Schreiber, K. (2021). *Accelerated neuromorphic cybernetics* (Ph.D. thesis). Universität Heidelberg.

Schreiber, K., Wunderlich, T., Spilger, P., Billaudelle, S., Cramer, B., Stradmann, Y., et al. (2022). Insectoid path integration on accelerated neuromorphic hardware.

Spilger, P., Müller, E., Emmel, A., Leibfried, A., Mauch, C., Pehle, C., et al. (2020). "hxtorch: PyTorch for BrainScaleS-2-perceptrons on analog neuromorphic hardware," in *IoT Streams for Data-Driven Predictive Maintenance and IoT, Edge, and Mobile for Embedded Machine Learning* (Cham: Springer International Publishing), 189–200.

Stone, T., Webb, B., Adden, A., Weddig, N. B., Honkanen, A., Templin, R., et al. (2017). An anatomically constrained model for path integration in the bee brain. *Curr. Biol*. 27, 3069–3085. doi: 10.1016/j.cub.2017.08.052

Stradmann, Y., Billaudelle, S., Breitwieser, O., Ebert, F. L., Emmel, A., Husmann, D., et al. (2021). Demonstrating analog inference on the BrainScaleS-2 mobile system. *arXiv preprint*.

Tkačik, G., Mora, T., Marre, O., Amodei, D., Palmer, S. E., Berry, M. J., et al. (2015). Thermodynamics and signatures of criticality in a network of neurons. *Proc. Natl. Acad. Sci. U.S.A*. 112, 11508–11513. doi: 10.1073/pnas.1514188112

Weis, J., Spilger, P., Billaudelle, S., Stradmann, Y., Emmel, A., Müller, E., et al. (2020). "Inference with artificial neural networks on analog neuromorphic hardware," in *IoT Streams for Data-Driven Predictive Maintenance and IoT, Edge, and Mobile for Embedded Machine Learning* (Cham: Springer International Publishing), 201–212.

Williams, S. R., and Stuart, G. J. (2002). Dependence of EPSP efficacy on synapse location in neocortical pyramidal neurons. *Science* 295, 1907–1910. doi: 10.1126/science.1067903

Wunderlich, T., Kungl, A. F., Müller, E., Hartel, A., Stradmann, Y., Aamir, S. A., et al. (2019). Demonstrating advantages of neuromorphic computation: a pilot study. *Front. Neurosci*. 13, 260. doi: 10.3389/fnins.2019.00260

Wunderlich, T. C., and Pehle, C. (2020). Eventprop: Backpropagation for exact gradients in spiking neural networks. *arXiv preprint* arXiv:2009.08378. doi: 10.1038/s41598-021-91786-z

Zenke, F., and Gerstner, W. (2014). Limits to high-speed simulations of spiking neural networks using general-purpose computers. *Front. Neuroinform*. 8, 76. doi: 10.3389/fninf.2014.00076

Zoschke, K., Güttler, M., Böttcher, L., Grübl, A., Husmann, D., Schemmel, J., et al. (2017). "Full wafer redistribution and wafer embedding as key technologies for a multi-scale neuromorphic hardware cluster," in *2017 IEEE 19th Electronics Packaging Technology Conference (EPTC)* (Singapore: IEEE).

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.