POETS: A Parallel Cluster Architecture for Spiking Neural Network

Mahyar Shahsavari, Jonathan Beaumont, David Thomas, and Andrew D. Brown

***Abstract*—Spiking Neural Networks (SNNs) are known as a branch of neuromorphic computing and are currently used in neuroscience applications to understand and model the biological brain. SNNs could also potentially be used in many other application domains such as classification, pattern recognition, and autonomous control. This work presents a highly-scalable hardware platform called POETS, and uses it to implement SNN on a very large number of parallel and reconfigurable FPGA-based processors. The current system consists of 48 FPGAs, providing 3072 processing cores and 49152 threads. We use this hardware to implement up to four million neurons with one thousand synapses. Comparison to other similar platforms shows that the current POETS system is twenty times faster than the Brian simulator, and at least two times faster than SpiNNaker.**

***Index Terms*—Parallel distributed system, reconfigurable architecture, spiking neural networks.**

I. Introduction

Artificial Neural Networks (ANNs) are a flexible and robust computing means for solving complex problems. However due to frequent accesses to memory, it suffers from a memory bottleneck when running on the conventional hardware plat-forms [1]. Spiking Neural Networks (SNNs) use biologically plausible neuronal models, and are a promising approach for hardware implementation of neural networks with capability of overcoming inefficient memory accessing by having the processor unit next to the memory [2]-[4]. SNNs are potentially capable of modeling complex information processing in the brain, in addition to other potential applications in accelerators, robotic brains, low-power mobile processors, deep learning [5], [6], or Medtech [7], [8].

Several pure software SNN simulators have been developed, such as NEURON [9], NEST [10], or BRIAN [11], and these are widely used as research tools in the community of computational neuroscience. Although these tools have been used to train, model and simulate biologically plausible neuronal networks, they are faced with hardware performance constraints such as power, speed, flexibility, memory accessing latency. Consequently, simulation of large-scale networks requires explicitly parallel processing [12]. In recent years advancements in high performance computing have led to the development of several large-scale hardware platforms dedicated to SNN applications, known as neuromorphic architectures. The most widely known large-scale neuromorphic systems are SpiNNaker [2], IBM TrueNorth [4], NeuroGrid [13], and BrainScales [14] projects.

Because of the parallel nature of neural networks, these large-scale concurrent systems are more efficient for data communication and spike transport compared to conventional platforms. However, the drawback is the programming complexity for these parallel systems, plus the need for analog or mixed analog/digital also increases complexity. In this work, we present the POETS (Partial Ordered Event Triggered Systems) [15] machine as a route to SNN simulation; POETS is a computation platform using an event-driven parallel programming model, backed by a custom FPGA many-core platform. POETS uses concepts from graph theory to provide a programming abstraction that makes programming this concurrent system manageable. This abstraction splits problems or applications into graphs, with events captured as messages moving between nodes in the graph, with events implementing both control- and data-flow. This allows for a high degree of concurrency and allows us to get very large numbers of CPUs to work closely together on a single application. The current POETS system consists of 48 FPGAs, providing 3072 processing cores and 49152 threads. Our contributions in this work are:

- investigating and explaining the POETS architecture, and how it is used to implement SNNs;
- hardware modeling of two neuron models, LIF (Leaky-Integrate-and-Fire) and Izhikevich, and a comparison of two models in a large-scale network;
- a demonstrator showing 4 million neurons, with each neuron connected to 1000 synapse, for a total of 4 billion synapses;
- a comparison between POETS and state-of-the-art simulators and large-scale platforms.

II. POETS Hardware Architecture

POETS is a project focusing on hardware support for an event-driven parallel programming model. Applications running on POETS must first be transformed into graphs, in which vertices construct computation units, while edges represent communication links which sending and receiving messages. Somewhat similar programming models are Google's Pregel model [16] and GraphStep model [17], which provide a computing abstraction using both synchronous and asynchronous way of passing the messages;

Manuscript received December 25, 2019; revised November 11, 2020. This work is supported by the UK Engineering and Physical Sciences Research Council under EPSRC grant EP/N031768/1.

M. Shahsavari, J. Beaumont and D. Thomas are with the Department of Electrical and Electronic Engineering, Imperial College London, UK (Corresponding author: M. Shahsavari; e-mail: m.shahsavari@imperial.ac.uk).

A. D. Brown is with School of Electronics and Computer Science, University of Southampton, UK.

doi: 10.18178/ijmlc.2021.11.4.1048

while the synchronous computing approach is deadlock-free and generally easier to program, the asynchronous approach can enable huge parallelism and scalability.

Efficient communication is a major strength of FPGA platforms, mainly due to an ability to process network traffic with minimal latency overheads. However, a major impediment to the wider adoption of FPGA platforms is the level of knowledge that is needed to develop in HDL (Hardware Description Language) effectively. Therefore, a promising solution is to provide a compiler/interpreter for FPGA developers to be able to use a higher level of abstraction for programming without taking the FPGA programming into consideration.

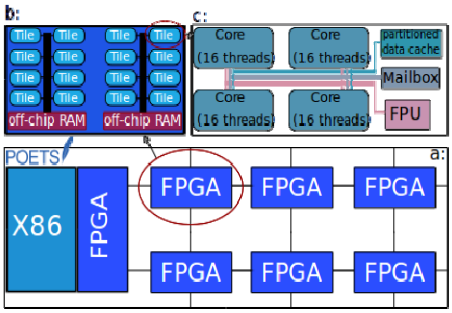


Fig. 1. a) One POETS box, each box has 7 FPGAs, 6 workers connected to an X86 GPP via one intermediate FPGA. b) A default configuration of the Tinsel Network on Chip (NoC) on a single FPGA. c) Default structure of a Tinsel tile; the cores are highly reconfigurable soft-processors using the Tinsel micro-architecture.

The POETS architecture currently consists of 8 boxes, where each box has an x86 server and 7 interconnected DE5-Net FPGA boards, as shown in Fig. 1a. Each DE5-Net supports 2×4*GB* DDR3 DRAMs, 4×8*MB* QDRII+ SRAMs, and 4×10*G* SFP+ ports. One FPGA is used as a PCIe to SFP+ bridge board, providing a fast connection between the x86 and remaining six worker FPGAs (Fig. 1a). The system supports both asynchronous and synchronous message exchanges between the cores. Each FPGA DE5-Net has 16 tiles, with each tile having 4 cores, and each core having 16 threads, so there are 1024 threads in each FPGA.

The computational heart of the POETS hardware is called Tinsel, which is a multithreaded RISC-V architecture optimised for FPGA implementation. When a thread executes an instruction with high latency, it is suspended and will resume after instruction completion. The system is highly-pipelined and parallel, using a design-time specific number of threads per core. The default number of threads per core is 16 and it is extensible to 32. Tinsel uses a 2D tiled network-on-chip (NoC), with each tile containing a single mailbox and some number of cores (4 by default), one FPU (floating-point unit), and caches. A data cache is used to access off-chip memory on each DE5, providing access to the two DDR3 DRAMS and four QDRII+ SRAMS (see Fig. 1b and Fig. 1c). If we assume a memory access occurs every four instructions, a single DDR3 DRAM can satisfy a maximum of 64 cores (32-bit) running at 250MHz. As is shown in Fig. 1c, a mailbox mechanism is used to send and receive the messages between the soft-processors. Mailboxes are connected together to establish a distributed network which a thread can send a message to any other threads. FPU operations are executed using Altera IP blocks and have 14 cycles latencies at 250MHz [18]. More detailed information including messaging, codding, interconnection, resource utilization for the Tinsel can be found in [19].

III. Model of Neurons

We will now discuss the two SNN model which we have mapped into the POETS system.

A. Leaky Integrate-and-Fire Model

LIF models are fast to simulate, and particularly attractive for large-scale network simulations [20]. Neurons integrate the spike inputs from other connected neurons, with each arriving input spike changing the internal potential of the neuron, known as neuron's membrane potential or state variable. When the integrated inputs cause the membrane potential to pass a threshold voltage, the action potential occurs – in other words, the neuron fires.

$$\tau_n \frac{dv}{dt} = -v(t) + RI_{syn}(t) \tag{1}$$

$$I_{syn}(t) = \sum_j g_{ij} \sum_n \alpha\left(t - t_j^n\right) \tag{2}$$

where: $v(t)$ represents the membrane potential at time t; $\tau_n = RC$ is the membrane time constant; and R is the membrane resistance. The total input current, $I_{syn}(t)$, is generated by the activity of pre-synaptic neurons. The total input current injected into a neuron is the sum over all current pulses, which is calculated in Equation 2. Time $t_j^{(n)}$ represents the time of the n_{th} spike of post-synaptic neuron j, and g_{ij} is the conductance of synaptic efficacy between neuron i and neuron j. Function $\alpha(t) = q.\delta(t)$, where q is the injected charge to the artificial synapse and $\delta(t)$ is the Dirac pulse function. If $I_{syn}(t)$ is big enough then the action potential can pass the threshold voltage, so the neuron fires. When there are no or only a few spikes in a time window, the neuron is in the leaky phase and the state variable decreases exponentially. The duration of this time window depends on τ_n= RC.

B. Izhikevich Model

Another well investigated simple model of neuron that has been simulated in our work is Izhikevich [21] model. Izhikevich model reproduces the physiological plausibility of Hodgkin-Huxley-type neuron yet are almost as computationally effective as the LIF neuron. The model is:

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I(t) \tag{3}$$

$$\frac{du(t)}{dt} = a(bv - u) \tag{4}$$

v is the membrane potential and I is the sum of the synaptic currents from different nodes connected to the neuron in Equation 3, whereas Equation 4 represents the u that is the membrane recovery variable. When v has reached its threshold then the neuron fires, and then reset happens according to Equation 4. The Izhikevich spiking model has the potential to generate several different firing patterns, which can be selected using four dimensionless parameters a, b, c, and d:

- a represents the time scale of the recovery variable u, where a smaller value means slower recovery;

$$v\left(v>v_{th}\right)=c,\ u\left(v>v_{th}\right)=u+d \tag{5}$$

- b represents the sensitivity of the recovery variable u to possible sub-threshold of the membrane potential v. Larger values indicated that v and u are strongly dependent on each other;
- c presents the reset value of v after spiking;
- d is the reset value of u after spiking.

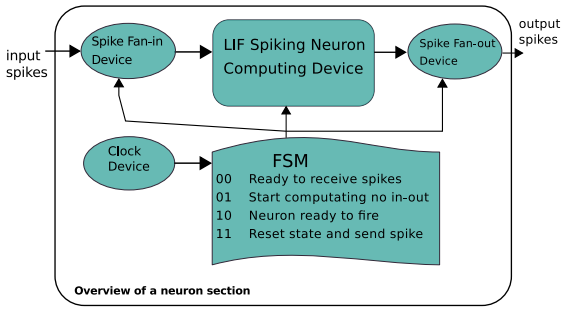


Fig. 2. An overview of a neuron section to be represented as a graph and read by compiler as an xml format file.

IV. Mapping Synchronous Problem on an Asynchronous Platform

The idea of the POETS system is to implement a highly scalable many-core system out of lots of tiny cores. Therefore, applications defining large numbers of simple devices with low computing complexity and a high degree of parallelism are desirable. Thus, this fact is considered in designing a neuron node consisting of four fan-in, fan-out, clock and computing devices. In this work, a finite state machine (FSM) is controlling the states of neuron synchronously, however the message propagation is performed asynchronously. An overview of neuron unit including different devices and states is depicted in Fig. 2. From a software developers point of view, the users and developer will not need to develop low-level FPGA code. A high-level python program is used to generate XML that represents the networks, the network elements, the number of neuron and connections from one to other neurons, according to parameters that are set by users. The POETS compiler will take care of the intermediate compilation and loading the network into an FPGA.

The aim of designing a neuromorphic architecture on POETS is to simulate large-scale SNNs in a reconfigurable, flexible and scalable platform. The sequential development procedure of the POETS compilation flow is shown in Fig. 3. Vertices in the network represent the neurons, and edges between vertices represent synaptic weight connections. A graph-based application called Graph Schema is used to create a network of neurons, with spike messages used to transfer data between neurons with a high degree of parallelism. The general approach is to decompose the graph into clusters of reconfigurable devices, where the amount of intra-cluster edges is large. Similar to the previous works for designing SNN on FPGA [22], [23], while also taking advantages of Tinsel for high speed access from FPGA cores to the memory (10 Gbps Ethernet MAC), the system is capable of large-scale networks simulation at high speed. After creating the network graph and the connections, an XML format file will be generated that is a representation of the network graph. This graph is then transferred to network instances and simulated on local or remote conventional GPP machines. The POETS compiler can also translate this file into FPGA-specific files which can be executed on Tinsel. Due to efficient data caching, in addition to the effective communication between threads via their mailbox, neuron devices in the bottom part of Fig. 3 receive and send messages synchronously via a high-speed network and transfer the neuron parameters and weight connection modifications to and from memory.

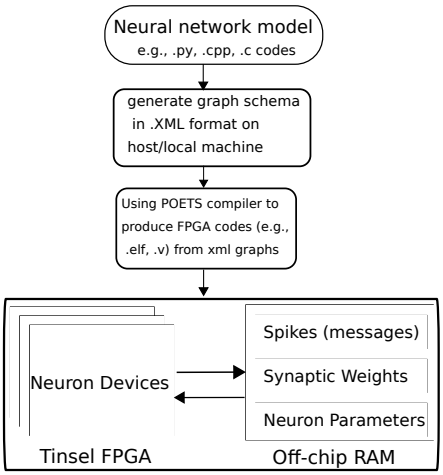


Fig. 3. An overview of the design flow, from high-level neural model down to implementation in FPGA.

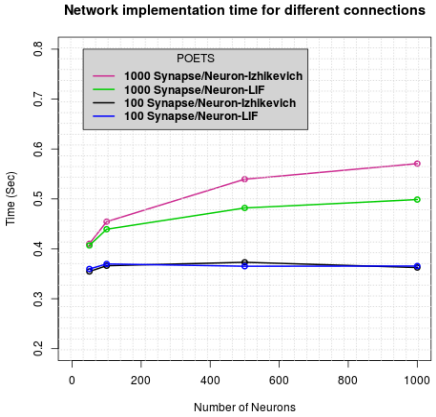


Fig. 4. A running time comparison of two LIF and Izhikevich models of neuron in POETS using small number of neurons with different numbers of synapses for each neuron.

V. Results

We have used this approach to model networks of neurons ranging from 50 to 500000 for one box, and up to 4 million neurons in 8 boxes. In all networks 20 percent of neurons are inhibitory neurons. Both the Izkikevich and LIF models of neuron have been used in the network, but the implementation costs (including running and mapping times) show little difference between Izhikevich or LIF model, as shown in Fig. 4. Another interesting result that can be extracted from the same data is the amount of parallelism. The hardware latency performance implementing anywhere from 50 to 1000 neurons is almost the same, as the nodes can be assigned to different threads, which compute simultaneously. More specific hardware characteristics are presented in Table I. Speed is a significant parameter has been evaluated in this platform, particularly when the number of neurons is increased. The maximum number of neurons that could be implemented on one box so far is 500

thousand. Therefore using 8 whole boxes, we are able to simulate 4 million neurons.

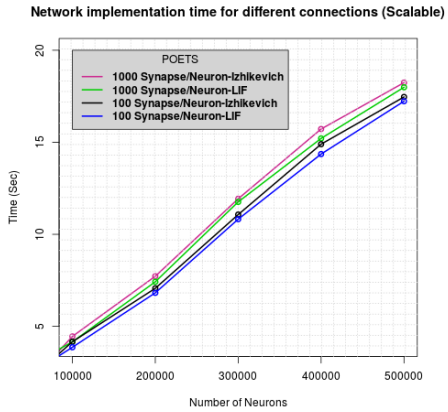


Fig. 5. A running time comparison of two LIF and Izhikevich models of neuron in POETS using scalable number of neurons with different number of synapses for each neuron.

For instance, the implementation time for 100000 neurons is only 4.05 second, however, this does not take into account compilation time, which could be longer where we use a conventional host computer. Running outputs for scalable devices are depicted in Fig. 5. We compare our outputs with the Brian simulator and the SpiNNaker hardware network in Fig. 6. The results demonstrate that poets is 20 times faster than Brian simulator version II. Compared to SpiNNaker, POETS is slightly slower for small networks, and more than two times faster for large neural networks.

TABLE I: CHARACTERISTICS OF ONE FPGA BOARD

FPGA Model	*DE5-Net*
Core	64
Threads	1024
DRAM	2 × 4GB DDR3
SRAM	4 × 8M B QDRII+
FPGA Clock Frequency	*250 MHz*
Power	<50 W

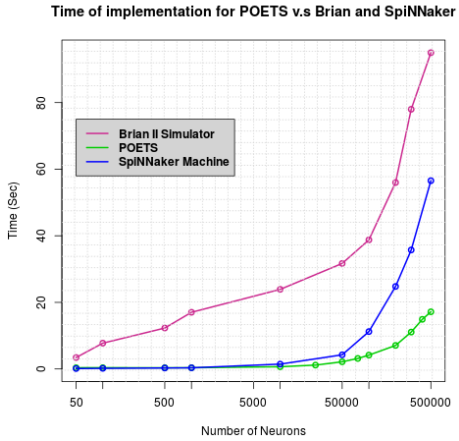


Fig. 6. Speed comparison of implementation among Brian simulator, SpiNNaker machine and POETS.

TABLE II: COMPARISON OF THE LARGE-SCALE NEUROMORPHIC SYSTEMS

Model/Properties	TrueNorth	Neurogrid	BrainScales	SpiNNaker	POETS
Technology	Digital	Analog	Analog	Digital	Digital (FPGA)
Feature size	28 nm	180 nm	180 nm	130 nm	28 nm
Chips	16	16	325	48	48
Power	3.2 W	3 W	500 W	80 W	42.8 w
Interconnect	2D mesh-unicast	Tree-multicast	Hierarchical	2D mesh-unicast	2D-mesh-unicast
Neuron model	Configurable LIF	Adaptive IF	Adaptive IF	Programmable	LIF/Izhikevich
Neurons	16 M	1 M	200 K	768 K	4 M
Synapses	4 G	4 G	40 M	768 M	4 G

VI. CONCLUSION

Spiking Neural Network is a promising approach for future computing platforms, with the ability of learning which could be used in three different scenarios:

- An accelerator in GPP platforms to overcome the Von Neumann memory bottleneck, for example in robotic brains, or low-power mobile processors, [24], [25].
- Direct implementation of spiking neural network on hardware, taking advantage of low-cost computing for the same purposes as ANN (e.g., Deep Learning) applications such as prediction, detection and recognition [5], [26].
- In the long term, understanding properties of biological neural networks could be used as a hippocampal prosthesis to be connected to the biological network or to replace a damaged biological memory, for example in the Alzheimer effected memory [8], [17].

Several large-scale spiking brain-like computing or neuromorphic hardware have been developed during recent years such as SpiNNaker, IBM TrueNorth, NeuroGrid and the BrainScales projects. In this work, we introduced POETS as a new large-scale neuromorphic system which is flexible using FPGA clusters, reliable with guaranty of receiving messages, and fast regarding to the parallel processing of data yet relatively low-power. The characteristics of large-scale systems are shown in the Table II to compare with POETS. In this work, we focused on scalability and architecture of system, while for future work we will investigate the accuracy of learning and neural network capabilities of POETS.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

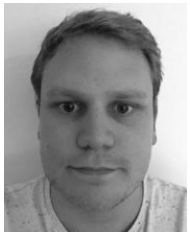
The authors' contributions can be listed as follows. The first author is responsible of output results, analyzing the data, running the tools and main writer of the article. The second author contributed in developing the tools, the third author's contributions are to develop tools, writing and revising the paper. Finally, the last author is responsible of the POETS project while has contributed in writing and revising this manuscript.

REFERENCES

[1] G. Indiveri and S.-C. Liu, "Memory and information processing in neuromorphic systems," *Proceedings of the IEEE*, vol. 103, no. 8, pp. 1379–1397, August 2015.

[2] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker Project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, May 2014.
[3] M. Shahsavari, P. Devienne, and P. Boulet, *Spiking Neural Computing in Memristive Neuromorphic Platforms*, Springer International Publishing, Cham, pp. 691–728, 2019.
[4] P. A. Merolla, J. V. Arthur, A. S. Cassidy *et al*., "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, August 2014.
[5] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Networks*, vol. 111, pp. 47–63, 2019.
[6] M. Fatahi, M. Ahmadi, A. Ahmadi, M. Shahsavari, and P. Devienne, "Towards an spiking deep belief network for face recognition application," in *Proc. 2016 6th International Conference on Computer and Knowledge Engineering (ICCKE)*, Oct. 2016, pp. 153–158.
[7] X.-J. Feng, E. Shea-Brown, B. Greenwald, R. Kosut, and H. Rabitz, "Optimal deep brain stimulation of the subthalamic nucleus - a computational study," *Journal of Computational Neuroscience*, vol. 23, no. 3, pp. 265–282, 2007.
[8] N. Kasabov, R. Schliebs, and H. Kojima, "Probabilistic computational neurogenetic modeling: From cognitive systems to alzheimer's disease," *IEEE Transactions on Autonomous Mental Development*, vol. 3, no. 4, pp. 300–311, 2011.
[9] N. T. Carnevale and M. L. Hines, *The NEURON Book*, Cambridge University Press, 2006.
[10] M.-O. Gewaltig and M. Diesmann, "Nest (neural simulation tool)," *Scholarpedia*, vol. 2, no. 4, p. 1430, 2007.
[11] D. F. M. Goodman, R. Brette, D. Goodman, and R. Brette, "Brian: A simulator for spiking neural networks in Python," *Frontiers in Neuroinformatics*, vol. 2, no. 5, 2008.
[12] M. Shahsavari, P. Devienne, and P. Boulet, "N2s3, a simulator for the architecture exploration of neuromorphic accelerators," in *Proc. 2nd International Workshop on Neuromorphic and Brain-Based Computing Systems (NeuComp 2015) in DATE Conference*, Grenoble, France, 2015.
[13] B. V. Benjamin, P. Gao, E. McQuinn *et al*., "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.
[14] J. Schemmel, D. Briiderle, A. Griibl, M. Hock, K. Meier, and S. Millner, "A wafer-scale neuromorphic hardware system for large-scale neural modeling," *IEEE*, pp. 1947–1950, May 2010.
[15] Poets project website. (2017). [Online]. Available: https://poets-project.org/
[16] G. Malewicz, M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, "Pregel: A system for large-scale graph processing," in *Proc. the 2010 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA, 2010, pp. 135–146
[17] M. DeLorimier, N. Kapre, N. Mehta *et al*., "Graphstep: A system architecture for sparse-graph algorithms," in *Proc. 14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, 2006, pp. 143–151.
[18] Altera floating-point megafunctions user guide. (2019). [Online]. Available: https://www.intel.co.jp/content/dam/altera-www/global/jaJP/pdfs/literature/ug/ug altfp mfug.pdf
[19] M. Naylor, S. W. Moore, and D. Thomas, "Tinsel: A manythread overlay for fpga clusters," in *Proc. 29th International Conference on Field Programmable Logic and Applications (FPL)*, Sept. 2019, pp. 375–383.
[20] R. Brette, M. Rudolph, T. Carnevale *et al*., "Simulation of networks of spiking neurons: a review of tools and strategies," *Journal of Computational Neuroscience*, vol. 23, no. 3, pp. 349–398, December 2007.
[21] E. M. Izhikevich, "Simple model of spiking neurons," *Trans. Neur. Netw.*, vol. 14, no. 6, pp. 1569–1572, November 2003.
[22] K. Cheung, S. R. Schultz, and W. Luk, "A large-scale spiking neural network accelerator for FPGA systems," in *Artificial Neural Networks and Machine Learning*, A. E. P. Villa, W. D. P. Érdi, F. Masulli, and G. Palm, Eds. Berlin, Heidelberg, 2012, pp. 113–120.
[23] R. C. Wang, G. Cohen, K. Stiefel, T. Hamilton, J. Tapson, and A. van Schaik, "An FPGA implementation of a polychronous spiking neural network with delay adaptation," *Frontiers in Neuroscience*, vol. 7, no. 14, 2013.
[24] M. Shahsavari and P. Boulet, "Parameter exploration to improve performance of memristor-based neuromorphic architectures," *IEEE Transactions on Multi-scale Computing Systems*, vol. 4, no. 4, pp. 833–846, Oct 2018.
[25] M. Walravens, E. Verreyken, and J. Steckel, "Spiking neural network implementation on fpga for robotic behaviour," *Lecture Notes in Networks and Systems*, vol. 96, pp. 694–703, 2020.
[26] E. Stromatias, D. Neil, F. Galluppi, M. Pfeiffer, S. Liu, and S. Furber, "Scalable energy-efficient, low-latency implementations of trained spiking deep belief networks on spinnaker," in *Proc. 2015 International Joint Conference on Neural Networks (IJCNN)*, July 2015, pp. 1–8.

Mahyar Shahsavari is a research associate at the Dept. of Electrical and Electronic Engineering, Imperial College London, United Kingdom. He achieved a PhD in computer science from CNRS, Lille University of Technology, where he developed a simulation platform for Spiking Neural Network using the model of a memristor nanodevice. His research interests include cognitive, neuromorphic and unconventional computing; machine learning applications running on novel architectures such as reconfigurable and parallel computing platform.



Jonathan Beaumont is a research associate at Imperial College London. He achieved a PhD in computer engineering from Newcastle University in 2018, where he researched asynchronous systems. Currently he works on the POETS project, developing software and applications to test a new experimental event-driven super-computing platform.

David B. Thomas is a senior lecturer in the Dept. of Electrical and Electronic Engineering at Imperial College London, where he performs research and teaching in the area of computer engineering, particularly at the boundaries between software, processors, and custom digital hardware. Much of his research concerns the efficient use of reconfigurable hardware, include the design of high-level languages and libraries for FPGAs, optimised numerical algorithms and IP cores, and the design of hardware optimised random number generators.

Andrew Brown is a professor of electronics with Southampton University, United Kingdom. He has held visiting posts at IBM Hursley Park (UK), Siemens NeuPerlach (Germany), Multiple Access Communications (UK), LME Design Auomation (UK), Trondheim University (Norway), Cambridge University (UK), and EPFL (Switzerland). He is a fellow of the IET and BCS, a chartered engineer, and a European engineer. He is a senior member of the IEEE.