Spike pattern recognition using artificial neuron and Spike-Timing-Dependent Plasticity implemented on a multi-core embedded platform

Filippo Grassia, Timothée Levi, E Doukkali, T Kohno

Spike pattern recognition using artificial neuron and Spike-Timing-Dependent Plasticity implemented on a multi-core embedded platform

F. Grassia[1*], T. Levi[2], E. Doukkali[1], and T. Kohno[3]

[1]LTI Lab., University of Picardie Jules Verne, France
[2]IMS Lab., University of Bordeaux, France
[3]IIS, Institute of Industrial Science, the University of Tokyo, Japan
(Tel: +33 03 23 50 36 90)

[*]filippo.grassia@u-picardie.fr

Abstract: The objective of this work is to use a multi-core embedded platform as computing architectures for neural applications relevant to neuromorphic engineering: e.g. robotics, artificial and spiking neural networks. Recently it has been shown how spike-timing-dependent plasticity (STDP) can play a key role in pattern recognition. In particular multiple repeating arbitrary spatiotemporal spike patterns hidden in spike trains can be robustly detected and learned by multiple neurons equipped with spike-timing-dependent plasticity listening to the incoming spike trains. This paper presents an implementation on a biological time scale of STDP algorithm to localize a repeating spatio-temporal spike patterns on a multi-core embedded platform.

Keywords: Spike-timing-dependent plasticity, spiking neural networks, pattern recognition, neuromorphic engineering.

1 INTRODUCTION

In the last years, a few hardware-based Spiking Neural Networks (SNN) systems were developed and the description of those pioneer platforms have gained remarkable attention [1-2]. On account of their parallel and distributed structures, spiking neuronal networks can simulate neuronal activities, potentially realizing an extremely large-scale network comparable to that of the human brain in future. Neuromorphic engineering aims to design SNN which will be used in Neuroscience for simulating the brain signal processing.

A second approach in the neuromorphic community concerns neuromimetic systems [3], which mimic more precisely the activity of biological cells and could replace the living part [4]. A neuromorphic system facilitates the building of a hybrid network incorporating both silicon and biological neurons. In recent times, the term neuromorphic has been used to describe analog, digital, mixed-mode analog/digital VLSI, and software systems that implement biologically realistic neural network models, from the electrophysiology of one single neuron to network plasticity rules. The analog circuit implementation consumes low power (down to nanowatts) per silicon neuron. However, is required to solve the problems induced by fabrication mismatch and temperature dependence to construct a large-scale network [5]. On the other hand, digital circuit implementation solves this limitation because it is far less sensitive to these factors, though power consumption tends to be higher than the analog circuit implementations. It is observed that the choice between analog and digital neural networks is application dependent. The architecture of SNN platforms will finally be a compromise between the computational cost and the model complexity which also constraints the achievable network size [6].

Different approaches to simulate spiking neural networks are to use either analog/digital VLSI or general purpose computing architectures [7] like clusters of CPUs or GPUs. While software tools can be configured for different types of models, hardware-based SNNs are dedicated to a given type of model.

A promising approach, that is a good trade-off, is the use of mixed hardware/software platform as the Parallella board by Adapteva [8]. This platform is designed for developing and implementing high performance, parallel processing applications developed to take advantage of the on-board Zynq programmable Soc and the Epiphany chip. The Epiphany 16-core chips consist of a scalable array of simple RISC processors programmable in C/C++ connected together with a fast on-chip network within a single shared memory architecture. The advantage of this mixed hardware/software platform is the presence of FPGA that offers a significant speedup over software designs, as well as size, weight, and power efficiencies.

The objective of this work is to use the Parallella board as computing architectures for neural applications relevant to neuromorphic engineering: e.g. robotics, artificial and spiking neural networks.

In particular; we present an implementation of the spike-timing-dependent plasticity (STDP) algorithm to localize a repeating spatio-temporal spike patterns hidden in spike trains on the Parallella board. This implementation have been realized for computation purposes taking into account biological time scale.

2 MATERIAL AND METHODS

In neuroscience, synaptic plasticity is the ability of the connection, or synapse, between two neurons to change in strength or efficacy in response to either use or disuse of transmission at preexisting synapses. Since memories are postulated to be represented by vastly interconnected networks of synapses in the brain, synaptic plasticity is one of the important neurochemical foundations of learning and memory. Spike-timing-dependent plasticity is a biological process that adjusts the strength of connections between neurons in the brain. The process adjusts the connection strengths based on the relative timing of a particular neuron's output and input action potentials (or spikes).

Recently it has been shown how STDP could play a key role by detecting repeating patterns and generating selective response to them. The concept of STDP has been shown to be a proven learning algorithm for forward-connected artificial neural network in pattern recognition.

In particular in the work presented by Masquelier [9-10], it has been shown that multiple repeating arbitrary spatiotemporal spike patterns hidden in spike trains can be robustly detected and learned by multiple neurons equipped with spike-timing-dependent plasticity (STDP) listening to the incoming spike trains (**Fig. 1**). The neurons become selective to successive coincidences of the patterns.

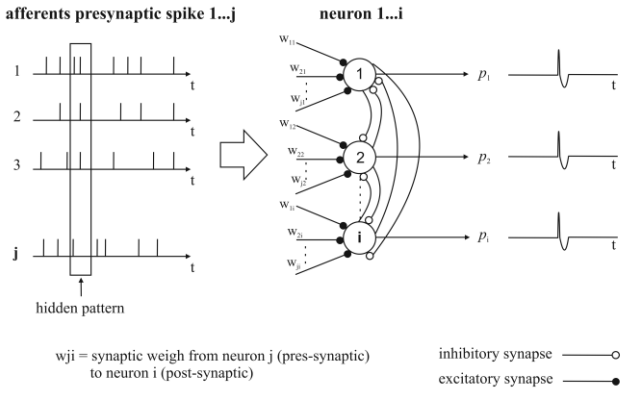

Fig. 1. Architecture of a neuronal network listening incoming spike trains with hidden patterns

In this work we will use the models presented in the work of Masquelier [10]. Furthermore, in the implementation on the multi-core embedded platform we will take into account biological time scale.

2.1 Neuron model

Specifically, for the neuron model, Gerstner's spike response mode (SRM) [11] is used. This model represents an alternative formulation to the integrate-and-fire model. Instead of defining the evolution of the neuron's membrane potential by a differential equation, SRM uses a kernel-based method to model the effect of spikes on the membrane potential (1).

At any time, the membrane potential is

$$p = \eta(t - t_i) + \sum_{j/t_j > t_i} w_{ji} \cdot \varepsilon(t - t_j) + \sum_{k/t_k > t_i} \mu(t - t_k) \qquad (1)$$

where the w_{ji} are the excitatory synaptic weights, between 0 and 1 (arbitrary units). Variation of synaptic weight w_{ji} from neuron j (pres-synaptic) to neuron i (post-synaptic), as function of the interval $\Delta t = t_i - t_j$. When a post-synaptic spike arises after a pre-synaptic spike ($\Delta t > 0$), the connection is reinforced (long-term potentiation (LTP), $\Delta Wji > 0$), whereas in the opposite case it is weakened (long-term depression (LTD)). The change of the synapse plotted as a function of the relative timing of pre- and postsynaptic action potentials is called the STDP function or learning window and arise between synapse types. In the equation (1) each presynaptic spike j, with arrival time t_j, is supposed to add to the membrane potential an excitatory postsynaptic potential (EPSP) of the form:

$$\varepsilon(t - t_j) = K \cdot \left(\exp\left(-\frac{t - t_j}{\tau_m} \right) - \exp\left(-\frac{t - t_j}{\tau_s} \right) \right) \cdot \Theta(t - t_j) \qquad (2)$$

where τ_m is the membrane time constant (here 10ms), τ_s is the synapse time constant (here 2.5ms), and Θ is the Heaviside step function and K is a multiplicative constant chosen so that the maximum value of the kernel is 1.

In the equation (1) the last emitted postsynaptic spike i has an effect on the membrane potential modeled as follows:

$$\eta(t - t_i) = T \cdot \left(K_1 \cdot \left(-\frac{t - t_i}{\tau_m} \right) - K_2 \cdot \left(\exp\left(-\frac{t - t_i}{\tau_m} \right) + \right. \right. \\ \left. \left. - \exp\left(-\frac{t - t_i}{\tau_s} \right) \right) \right) \cdot \Theta(t - t_i) \qquad (3)$$

where T is the threshold of the neuron (here 550, arbitrary units) and $K_1 = 2$, $K_2 = 4$ are constants. Furthermore in the equation (1), when a neuron fires at time t_k, it sends to the others an inhibitory postsynaptic potential (IPSP) of the form:

$$\mu(t - t_k) = -\alpha \cdot T \cdot \varepsilon(t - t_k) \quad (4)$$

ε, η, and μ kernels were rounded to zero when, respectively, $t - t_j$, $t - t_i$, and $t - t_k$ were greater than $7\tau_m$.

2.2 Spike-Timing-Dependent Plasticity rule

STDP rules are the most common form of learning used in SNN. The dynamics of a SNN and the formation of its connectivity are governed by synaptic plasticity. Plasticity rules formulate the modifications which occur in the synaptic transmission efficacy, driven by correlations in the firing activity of pre- and postsynaptic neurons. At the network level, spikes are generally processed as events, and the synaptic weight w_{ji} (connection from neuron j to neuron i) varies over time, according to the learning rules. As done in the work of Masquelier [10] we used an additive exponential update rule of the form (5) for the implementation on a multi-core embedded platform.

$$\Delta(w_j) = \begin{cases} a^+ \cdot \exp\left(\dfrac{t_j - t_i}{\tau^+}\right) & \text{if } t_j \le t_i \quad \text{(LTP)} \\ -a^- \cdot \exp\left(-\dfrac{t_j - t_i}{\tau^-}\right) & \text{if } t_j > t_i \quad \text{(LTD)} \end{cases} \quad (5)$$

where $a^+ = 0.03125$ and $a^- = 0.85 \cdot a^+$, $\tau^+ = 16.8$ms and $\tau^- = 33.7$ms. We implemented a restricted learning window respectively of $[t_i - 7 \cdot \tau^+, t_i]$ for LTP and $[t_i , t_i + 7 \cdot \tau^-]$ for LTD.

2.3 Computing Architecture

SNNs fall into the third generation of neural network models, increasing the level of realism in a neural simulation. The idea is that neurons in the SNN do not fire at each propagation cycle (as happens with typical multi-layer perceptron networks), but rather, only fire when a membrane potential reaches a specific value. When
a neuron fires, it generates a signal that travels to other neurons which, in turn, increase or decrease their potentials in accordance with this signal. Computational neuroscience commonly relies on software-based processing tools (NEURON, NEST, PCSIM, Brian, etc.).

As mentioned in the introduction, neuromorphic engineering is a new interdisciplinary field that takes inspiration from biology, physics, mathematics, computer science and engineering to design analog, digital, and mixed-mode analog/digital VLSI and software systems to mimic neuro-biological architectures present in the nervous system.

Some of these platforms are dedicated to the simulation of SNNs, and take into account the timing of input signals by precisely computing the neurons' asynchronous spikes. While software tools can be configured for different types of models [12], hardware-based SNNs are dedicated to a given type of model.

In this work we use the Parallella board a mixed hardware/software platform by Adapteva [8] to investigate the capability of simulating a STDP algorithm to localize a repeating spatio-temporal spike patterns on a biological time scale.

The Parallella board comes with either a 16 core or a 64 core Epiphany chip (here we used the 16 core version), it contains a Zynq SOC (FPGA + ARM A9) and an Epiphany coprocessor which are connected through the eLink interface and AXI bus (**Fig. 2**). It can be connected through a Gigabit Ethernet port. It also contains a μHDMI port as well as 2 μUSB ports and a μSD slot.

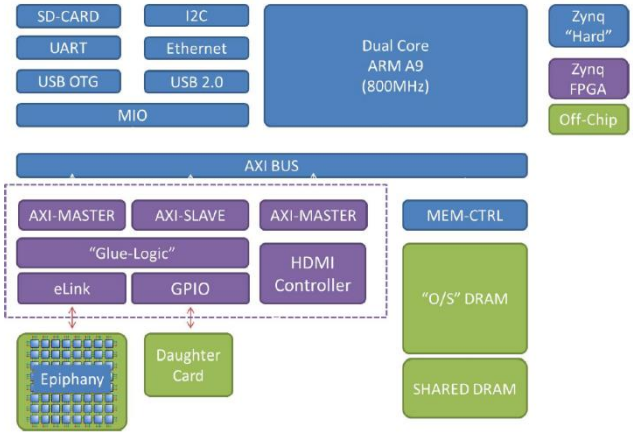

Fig. 2. Parallella High Level Architecture. This figure was taken from Parallella reference manual [8]

The presence of the FPGA offers a significant speedup over software designs, as well as size, weight, and power efficiencies. In the provided bitstreams, the FPGA is configured to drive the HDMI, GPIO pins and the host-side eLink communication between the ARM and Epiphany chip.

A single epiphany core, is computationally similar to the ARM. Both are 32-bit RISC cores capable of efficient floating-point and discrete operations.

The ARM is capable of running a full Operating System, such as Linux and can interface with the external world via the FPGA.

Furthermore, we make use of the Epiphany Software Development Kit (eSDK) that is a software development environment targeting the Epiphany multicore architecture. The eSDK is based on standard development tools including an optimizing C-compiler, functional simulator, debugger, and multicore integrated development environment (IDE). Its drawback is its limitation of connected cores at 4096, which limits the capabilities in terms of simulating a sizable part of the human brain.

3 NETWORK ARCHITECTURE

3.1 SNN execution architecture on the Parallella platform

The spike pattern recognition algorithm was implemented on a shared memory multi-core embedded platform, with single program, multiple data (SPMD) as a technique employed to achieve parallelism. Tasks are split up and run simultaneously on multiple processors (here 16 core) with different input lines in order to obtain results faster.

The implementation was realized in C language using the eSDK and compiler provided by Adapteva. We compiled the code with gcc for the ARM and with egcc, a modified compiler, for the Epiphany chip. To evaluate the speedup of the multi-core platform, the algorithm was implemented both on a single core ARM and on the Epiphany coprocessor mediated by the ARM host.

In particular, the algorithm was evaluated for 2048 inputs spike trains (128 inputs per core), with two hidden patterns integrated in parallel by five downstream SRM neurons, through excitatory synapses governed by STDP. Lateral inhibitory connections are set up between the SRM neurons, so that as a neuron fires, it sends an inhibitory postsynaptic potential (IPSP) of the form presented in the equation (4) to its neighbors.

This implementation have been realized for computation purposes taking into account biological time scale. The inputs spike trains were generated for a simulation time of 500 seconds and stored in the shared memory. Anyway, the platform provide an FPGA that can be configured to drive GPIO pins for data transfer.

As mentioned before, our interest is in the systems with a biological time scale. Thus, we choose a computation time of 1 millisecond, and then our architecture has to compute the value of all parameters within that time. The epiphany has very limited external IO capabilities.

In fact, any spike train data into or out from the coprocessor needs to be mediated by the ARM host, final results are all controlled by applications running on the ARM.

3.2 Data generating process for the network

In neuroscience, the words firing and spiking commonly refer to action potentials generated by a neuron. Simulating input spike trains like the ones in the raster plot as shown in **Fig. 3** requires only one piece of information: the firing rate of the neuron. As used in the work of Masquelier [10] we generated spikes independently using a Poisson process with a variable instantaneous firing rate that varies randomly between 0 and 90 Hz. The maximal rate change was chosen so that the neuron could go from 0 to 90 Hz in 50 ms.

Finally a part the spike trains, defined as the ‘pattern’ to be repeated, was replaced for half input lines into sections of 50ms. We randomly pick one of these sections and copy the corresponding spikes. As shown in **Fig. 3**, we generate two different hidden patterns (red circles, green circles) that repeat at random intervals within stochastic Poissonian activity (blue circles).

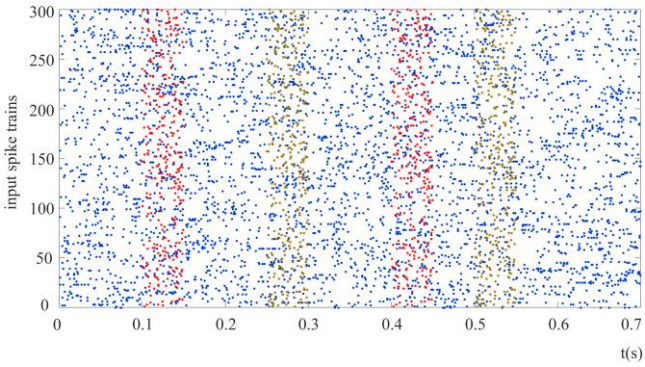

Fig. 3. Spatiotemporal spike patterns. In consequence of space constraints, we show only the first 300 input (out of 2048)

4 RESULTS

The spike pattern recognition algorithm was simulated for 500 seconds on the host ARM. We explored a situation that focuses on the recognition of two previously unknown patterns integrated in parallel by five downstream SRM neurons, through excitatory synapses governed by STDP and lateral inhibitory connections.

As found in the work of Masquelier [10] we found that the same neuron cannot become selective to two distinct patterns and inhibition encourages the neurons to distribute themselves across all the patterns. The simulation time of about 13 minutes shows that real-time simulation cannot be reach with only one ARM core.

Pursuant to our goal of implementing the algorithm in the biological time scale we run the simulation using the 16 cores Epiphany chip. In this example patterns 1 and 2 were learned by four neurons, and one neurons stopped firing after too many spikes had generated outside the patterns (they did not learn any pattern). **Fig. 4** shows a typical result. Here we plotted the membrane potential as a function of simulation time, at the end of the simulation. After about 80 pattern presentations and 600 spikes' generation, selectivity to the pattern is emerging: gradually the neuron almost stops spiking outside the pattern, while it does spike most of the time when the pattern is present. As shown in **Fig. 4** neuron 1 and neuron 2 become selective to the pattern 1 and 2 respectively. The measured postsynaptic spike latency is about 5ms and there are no false alarms after the 676th spike that is for the last 400s of simulation. The maximum speedup measured on the platform was 14 times.

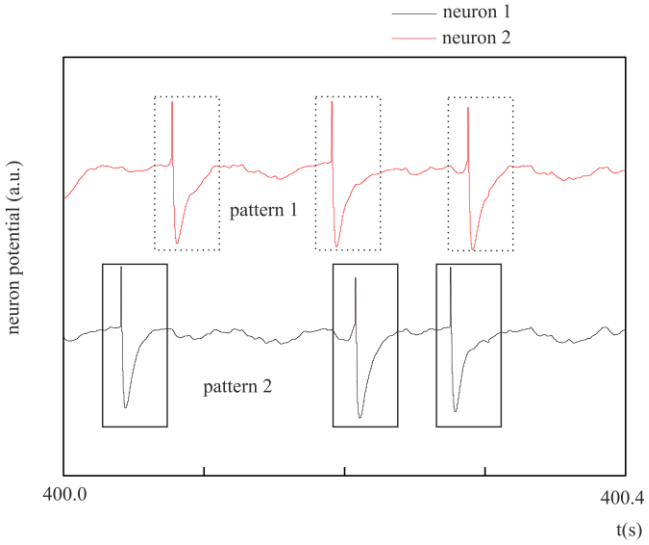

Fig. 4. The membrane potential is plotted as a function of time for the neuron 1 and 2. Neuron 3 and 4 (not shown here) become selective to the pattern 1 and 2 respectively. The neuron 5 did not learn any pattern

5 CONCLUSION

In the present work we presented an implementation on the Parallella board of STDP algorithm to localize a repeating spatio-temporal spike patterns hidden in spike trains. This implementation have been realized taking into account biological time scale using the 16 cores Epiphany chip. The Parallella board provides a large number of cores that are able to perform floating point operations, which makes it suitable for neural applications relevant to neuromorphic engineering as for example pattern recognition. The maximum speedup measured in this work shows how real-time simulation can be reached for "small neural network" using Epiphany coprocessor mediated by the ARM host. This calculation was four times faster than the biological real-time. We plan to use this hardware and software platform to improve the hybrid technique, also called "dynamic-clamp" that consists of connecting artificial and biological 'in vivo' or/and 'in vitro' neurons to study the function of neuronal circuits using microelectrode arrays.

ACKNOWLEDGMENTS

This work was financially supported by the "PHC Sakura" program (project number: 35966TL), implemented by the French Ministry of Foreign Affairs, the French Ministry of Higher Education and Research and the Japan Society for Promotion of Science. The authors would like to thank CAPROM association of Saint Quentin for dissemination of results.

REFERENCES

[1] Merolla PA, Arthur JV, Alvarez-Icaza R, Cassidy AS, et al. (2014), A million spiking-neuron integrated circuit with a scalable communication network and interface. Science 345.6197 (2014): 668-673

[2] Kohno T, Li J, Aihara K (2014), Silicon neuronal networks towards brain-morphic computers. Nonlinear Theory and Its Applications, IEICE, 5(3), 379-39

[3] Levi T, Lewis N, Tomas J, Saighi S, et al.(2008), Neuromimetic Integrated Circuits, VLSI Circuits for Biomedical Applications. Iniewski Editor

[4] Le Masson G, Renaud-Le Masson S, Debay D, Bal T (2002), Feedback inhibition controls spike transfer in hybrid thalamic circuits. Nature, 417(6891), 854-858

[5] Grassia F, Buhry L, Lévi T, Tomas J, Destexhe A, Saïghi S (2011), Tunable neuromimetic integrated system for emulating cortical neuron models. Frontiers in neuroscience, 5, 134

[6] Grassia F, Lévi T, Tomas J, Renaud S, Saïghi S (2011), A neuromimetic spiking neural network for simulating cortical circuits. In Information Sciences and Systems (CISS), 2011 45th Annual Conference on (pp. 1-6). IEEE

[7] Furber S, Brown A (2009), Biologically-inspired massively-parallel architectures-computing beyond a million processors. In Application of Concurrency to System Design, ACSD'09. Ninth International Conference (pp. 3-12) IEEE

[8] Parallella, https://www.parallella.org/

[9] Masquelier T, Guyonneau R, Thorpe SJ (2008), Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains. PloS one, 3(1), e1377

[10] Masquelier T, Guyonneau R, Thorpe SJ (2009), Competitive STDP-based spike pattern learning. Neural computation, 21(5), 1259-1276

[11] Gerstner W, Kistler WM (2002), Spiking neuron models. Cambridge University Press

[12] Brette R, Rudolph M, Carnevale T, Hines M, et al. (2007), Simulation of networks of spiking neurons: A review of tools and strategies. Journal of Computat ional Neuroscience, 23:349–398