# Physical Mapping of Spiking Neural Networks Models on a Bio-inspired Scalable Architecture

J. Manuel Moreno[1], Javier Iglesias[2], Jan L. Eriksson[2], and Alessandro E.P. Villa[3]

[1] Technical University of Catalunya, Dept. of Electronic Engineering
Campus Nord, Building C4, c/Jordi Girona 1-3, 08034-Barcelona, Spain
moreno@eel.upc.edu
[2] Laboratory of Neuroheuristics, Information Systems Department INFORGE
University of Lausanne, Lausanne, Switzerland
Javier.Iglesias@unil.ch, jan@lhn.unil.ch
[3] INSERM U318, University Joseph-Fourier Grenoble 1, Pavillon B
CHUG Michallon, BP217, F-38043 Grenoble Cedex 9, France
Alessandro.Villa@ujf-grenoble.fr

**Abstract.** The paper deals with the physical implementation of biologically plausible spiking neural network models onto a hardware architecture with bio-inspired capabilities. After presenting the model, the work will illustrate the major steps taken in order to provide a compact and efficient digital hardware implementation of the model. Special emphasis will be given to the scalability features of the architecture, that will permit the implementation of large-scale networks. The paper will conclude with details about the physical mapping of the model, as well as with experimental results obtained when applying dynamic input stimuli to the implemented network.

## 1 Introduction

Spiking neural networks models have attracted a considerable research interest during the last years [1], [2] because of their biological plausibility and their suitability for a physical hardware implementation. From the different learning mechanisms available for this neural models Spike Timing Dependent Plasticity (STDP) has received an increasing interest [3] because of experimental evidence [4] and observations suggesting that synaptic plasticity is based on discrete dynamics [5].

In this paper we shall consider a spiking neural network model whose learning mechanism is based on discrete variables [6]. After presenting the model the sequence of steps driving to its physical realization will be explained. Then the implementation on the model on a scalable hardware architecture with bio-inspired features will be described. The implementation results show that it is possible to attain real-time processing capabilities for dynamic visual stimuli.

## 2 Spiking Neural Network Model

The model consists of Leaky Integrate-and-Fire neuromimes connected by synapses with variable weight depending on the time correlation between pre- and post-synaptic

S. Kollias et al. (Eds.): ICANN 2006, Part I, LNCS 4131, pp. 936 – 943, 2006.

spikes. The synaptic potentials are added until their result $V_i(t)$ overcomes a certain threshold. Then a spike is produced, and the membrane value is reset. The simplified equation of the membrane value is:

$$V_i(t+1) = \begin{cases} 0 & when\ S_i(t) = 1 \\ k_{mem} \cdot V_i(t) + \sum J_{ij}(t) & when\ S_i(t) = 0 \end{cases} \tag{1}$$

where $k_{mem}=exp(-\Delta t/\tau_{mem})$, $Vi(t)$ is the value of the membrane and $S_i(t)$ is the state variable which signals the occurrence of a spike. The value of $J_{ij}$ is the output of each synapse *(ij)* where *j* is the projecting neuron and *i* is the actual neuron.

When a spike occurs in the pre-synaptic neuron, the actual value of the synaptic output $J_{ij}$ is added to the weight of the synapse multiplied by an activation variable *A*. Conversely, if there is no pre-synaptic spike then the output $J_{ij}$ is decremented by a factor $k_{syn}$. Then, the value of $J_{ij}$ corresponds to the following equation:

$$J_{ij}(t+1) = \begin{cases} J_{ij}(t) + (w_{RiRj} \cdot A_{RiRj}(t)) & when\ S_j(t) = 1 \\ k_{syn} \cdot J_{ij}(t) & when\ S_j(t) = 0 \end{cases} \tag{2}$$

where *j* is the projecting neuron and i is the actual neuron. *R* is the type of the neuron : excitatory or inhibitory, *A* is the activation variable which controls the strength of the synapse, and $k_{syn}$ is the kinetic reduction factor of the synapse. If the actual neuron is inhibitory, this synaptic kinetic factor will reset the output of the synapse after a time step, but if the actual neuron is excitatory, it will depend on the projecting neuron. If the projecting neuron is excitatory the synaptic time constant will be higher than if it is inhibitory. The weight of each synapse also depends on the type of neuron it connects. If the synapse connects two inhibitory neurons, the weight will always be null, so an inhibitory cell cannot influence another inhibitory cell. If a synapse is connecting two excitatory neurons, it is assigned a small weight value. This value is higher for synapses connecting an excitatory neuron to an inhibitory one, and it takes its maximum value when an inhibitory synapse is connected to an excitatory cell.

The changes in strength of an excitatory-excitatory synapse depend on the variable *A* which is a function of on an internal variable $L_{ij}$ given by the following equation:

$$L_{ij}(t+1)=k_{act} \cdot L_{ij}(t) + (YD_j(t) \cdot S_i(t)) - (YD_i(t) \cdot S_j(t)) \tag{3}$$

where $k_{act}$ is a kinetic activity factor, which is the same for all the synapses and *YD* is a “learning” decaying variable that depends on the interval between a pre-synaptic spike and a post-synaptic spike. When there is a spike, *YD* reaches its maximum value at the next time step. In the absence of a spike the value of *YD* will be decremented by the kinetic factor $k_{learn,}$ which is the same for all synapses. When a pre-synaptic spike occurs just before a post-synaptic spike, then the variable $L_{ij}$ is increased and the synaptic strength becomes larger, thus corresponding to a potentiation of the synapse. When a pre-synaptic spike occurs just after a post-synaptic spike, the variable $L_{ij}$ is decreased, the synaptic weight is weakened , thus corresponding to a depression of the synapse. For all kind of synapses, except the excitatory-excitatory, the activation variable is always is set to 1.

## 3 Hardware Implementation

In this section we shall consider the detailed implementation of the model, as well as its optimization for an efficient hardware realization.

The overall organization of the neuron model is depicted in Figure 1.

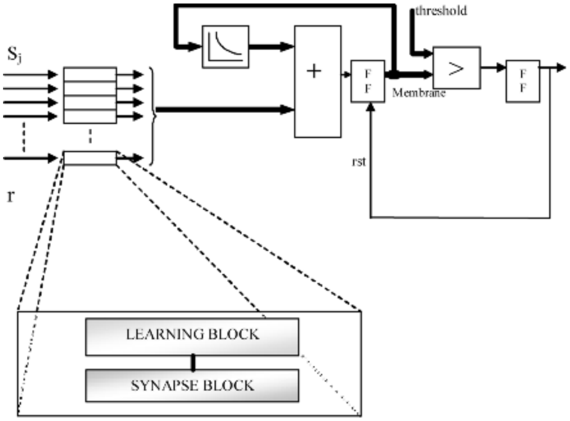

**Fig. 1.** Overall organization of the neuron model

The description of the neuron block can be divided in three main parts. In the first part the spikes(s) received from outside (probably from other neurons) are processed through a block that encompasses two additional sub-blocks, synapse and learning, which will be explained later. These sub-blocks are used to give appropriate inputs to the next building blocks of the neuron model.

In a second stage, the inputs are added or subtracted, depending on the nature (r) of the previous neuron (i.e. excitatory or inhibitory), to the decayed value of the membrane . The result of this final addition is what we call "membrane value" and it is stored in a flip-flop (FF in Figure 1). This membrane value is always processed through a decay function which gives the adding value in the next time step. The registered output of the membrane is compared in the third sub-block with a predefined threshold value. When the membrane value reaches this threshold, a spike is produced. This spike will be delayed in the final part with a flip-flop which models the refractory time. When finally the spike goes out from the neuron, it produces a reset (rst signal in Figure 1) in the flip-flop which stores the value of the membrane.

A major building block in the neuron model is the decay block, since it will be used both in the synapse and in the learning blocks. This block is aimed to implement a logarithmic decay of the input; it is obtained with a subtraction and controlling the time when it is done depending on the input value. The organization of this block is presented in Figure 2. In this figure the decaying variable is labeled *x*. A new value of *x* will be the input of a shift register which is controlled by the most significant bit (*MSB*) of *x* and by an external parameter *mpar*. The output of this shift register will be subtracted from the original value of *x*. This operation will be done when the time control indicates it. The time control is implemented by the value of a counter that is compared with the result of choosing between the external value *step* and the product *(MSB–mpar)·step*. The decay variable $\tau$ depends on the input parameters *mpar* and *step*.that is controlled by the time when it is done depending on the input value.

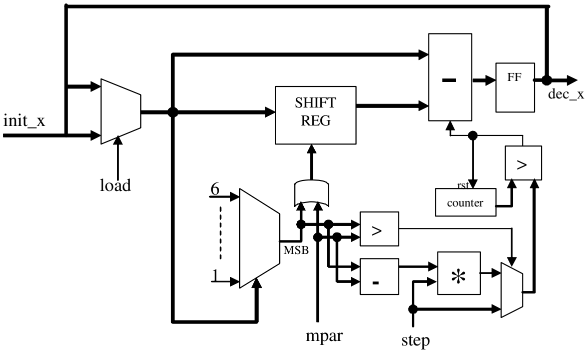

**Fig. 2.** Organization of the decay block

The learning block "measures" the interval between a spike in the projecting neuron *j* and the actual neuron *i*. Depending on these timings and the types of the two neurons, the synaptic strength will be modified. When a spike is produced by the projecting neuron, the variable *YD* is set to its maximum value and starts to decay. If a spike is produced by the actual neuron immediately after the presynaptic neuron the value of $YD_j$ is added to the decaying value of *L*. Conversely, if a spike is produced at first in the actual neuron and later in the projecting neuron, then the value of $YD_i$ is subtracted to the decaying value of *L*. If the *L* variable overcomes a certain threshold $L_{th}$, positive or negative, then the activation variable *A* is increased or decreased, respectively, unless the variable had reached its maximum or minimum, respectively. If the variable *A* is increased, then *L* is reset to the value $L\text{-}2\cdot L_{th}$; if *A* is decreased, then *L* is reset to $L+2\cdot L_{th}$. Figure 3 illustrates the organization of the learning block.

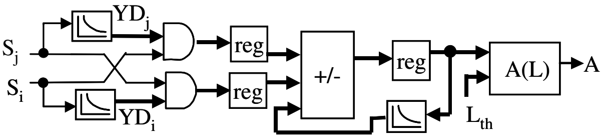

**Fig. 3.** Organization of the learning block

The synapse block is aimed to set the value of *J* (analogous to the the sum of all post-synaptic membrane potentials) and depends on four factors: the activation level *A* of the synapse, the spiking state of the projecting neuron $S_j$ and the types of the pre- and post-synaptic neurons ($R_i$ and $R_j$).

A given weight is set for each synapse. This weight is multiplied by the activation variable *A* by means of a shift register, such that if *A*=0, the weight is multiplied by 0, if *A*=1 it is multiplied by 1, if *A*=2 it is multiplied by 2, and if A=3 it is multiplied by 4. This weighted output is added to the decaying value of the variable *J*.

This operation depends on the neuronal types ($R_i$ and $R_j$). In the current case study there are only two types of neurons, excitatory and inhibitory. If both neurons are inhibitory the weight of the synapse is set to 0 and the value of $J$ is always 0 and no decay is implemented. For the other three types of synapses the time constants are multiplexed, and the multiplexer is controlled by the types of neurons ($R_i$,$R_j$). The value of $J$ is obtained at the output of the decay block controlled by the multiplexer. Figure 4 shows the organization of the synapse block.

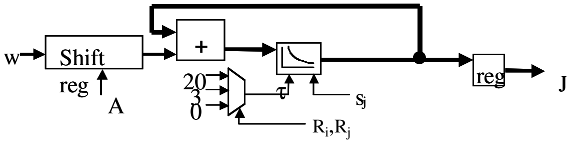

**Fig. 4.** Organization of the synapse block

The resolution required to represent the values of the variables and the number of operations to be performed may pose a serious limitation for the final implementation. Therefore, an important step consisted in evaluating the model and tuning its parameters in order to get a satisfactory performance. The implementation used in this study has been based on a neural network of size 15x15 with a connectivity pattern of 24 neurons corresponding to a neighborhood of 5x5. The distribution of the 20% inhibitory cells was random. The weights, *w*, and the initial activation variables, *A*, were also chosen randomly. Dynamic gradient stimuli have been applied to the neural network. A sequence of vertical bars of gradient intensity move over "strips" of neurons placed in the 2D array of the neural network.

The vertical bars may move at different speeds (i.e. spatial frequency). A neuron "hit" by the stimulus receives an input that is proportional to the gradient intensity. The activity of the network has been studied in a "training" condition and in a "test" condition. During training the spatial frequency of the stimulus has been incremented by discrete harmonics (2x, 4x, etc.) in one direction (the "forward" direction). During test, the stimuli were presented in both forward and reverse sense. A Gaussian noise (Mean 0, SD= 48) is applied to all neurons during all the time. The characteristics of the input applied to each neuron are the following:

- $T_{CLK}$: 20 ns. Maximum amplitude: 127.
- Training period: 20 µs. Forward sense
- Test period: 10 µs. Forward and Reverse sense

The results from this experiment demonstrate that the selected structure of our neural network is able to perform an implicit recognition of dynamic features based on simple unsupervised STDP rules.

In a first attempt to reduce the complexity of the final hardware implementation the resolution of the parameters has been reduced by 2 bits. By repeating the simulation experiments explained previously we could determine that this is the minimum accuracy required by the system in order to exhibit discrimination features for dynamic input stimuli. Table 1 shows the new values of the internal parameters after this optimization process.

**Table 1.** Resolution of the parameters for an optimized implementation

| Parameter | New value |
|---|---|
| Membrane resolution | 10 bits |
| Threshold | +160 |
| Input (J) resolution | 6 bits |
| Weights ($R_i$,$R_j$) (00, 01, 10, 11) | [0:8], [64:128], [128:256], [0:0] |
| YD resolution | 4 bits |
| L resolution | 6 bits |
| Membrane decay time constant | 20 |
| YD decay time constant | 20 |
| L decay time constant | 4000 |
| $J_{R_i,R_j}$ decay time constants ($R_i$,$R_j$) (00, 01, 10, 11) | (20, 0, 3, 0) |

Once this simplification has been performed a further simplification has been carried out [7] in the design of the constituent building blocks. In this optimization a serial approach has been used in order to keep the functional units as compact as possible.

## 4 Implementation on a Bio-inspired Architecture

The POEtic tissue [8] constitutes a flexible hardware substrate that has been specifically conceived in order to permit the efficient implementation of bio-inspired models. The tissue may be constructed as a regular array composed of POEtic chips, each of them integrating a custom 32-bit RISC microprocessor and a custom FPGA with dynamic routing capabilities.

The custom FPGA included in the POEtic chip is composed of a bi-dimensional array of elementary programmable elements, called molecules. Each molecule contains a flip-flop, a 16-bit lookup table (LUT) and a switchbox that permits to establish programmable connections between molecules.

After the optimization carried out on the neural model in order to facilitate its hardware realization it has been mapped on to the molecules that constitute the POEtic device. The molecule organization shown in Fig. 5 corresponds to the actual structure of the FPGA present in the POEtic device, which is arranged as an 8x18 array of molecules.

The VHDL models developed for the POEtic tissue have been configured and simulated to validate the functionality of the neuron model designed above. After this

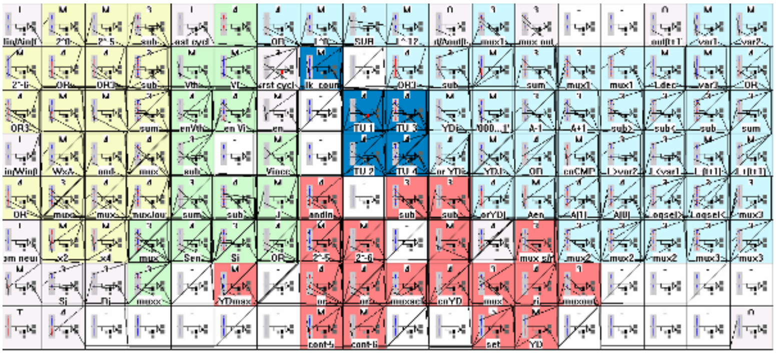

**Fig. 5.** Molecule-level implementation of the SNN model

validation stage the strategy for the simulation of large-scale SNN models has been considered. Since in its actual implementation the POEtic chip only allows for the implementation of a single neuron it will be necessary to use an array of POEtic chips whose functionality should be time–multiplexed in order to emulate the entire network. This means that every POEtic chip should be able to manage a local memory in charge of storing the weights and learning variables corresponding to the different neurons it is emulating in time.

A 16-neurons network organized as a 4x4 array has been constructed using this principle. This would permit the emulation of a 10,000-neurons network in 625 multiplexing cycles. Bearing in mind that each neuron is able to complete a time step in 150 clock cycles, this means that the minimum clock frequency required to handle input stimuli in real time (i.e., to process visual input stimuli at 50 frames/second) is around 5 MHz far within the possibilities of the actual clock frequency achieved by the POEtic tissue (between 50 MHz and 100 MHz).

The visual stimuli will come from an OmniVision OV5017 monochrome 384x288 CMOS digital camera. Specific VHDL and C code have been developed in order to manage the digital images coming from the camera. To test the application, artificial image sequences have been generated on a display and then captured by the camera for its processing by the network.

## 5 Conclusions

In this paper we have presented the detailed translation process of a biologically plausible spiking neural network model onto a physical hardware implementation based on a scalable architecture with bio-inspired features. During the translation process special attention has been paid to the accuracy constraints of the implementation, so as to obtain a compact physical realization. The results of the current implementation demonstrate that the proposed approach is capable of supporting the real-time needs of large-scale spiking neural networks models. Our current work is concentrated on the physical test and qualification of the POEtic chips received from the foundry using the development boards that have been constructed for the POEtic tissue. After that the configuration corresponding to the proposed model will be downloaded and physically tested on the actual chips.

## Acknowledgements

The work presented in this paper has been funded by the grant IST-2000-28027 (POEtic) of the European Community and by grant OFES 00.0529-2 of the Swiss government. The information provided is the sole responsibility of the authors and does not reflect the Community's opinion. The Community is not responsible for any use that might be made of data appearing in this publication.

## References

1. Maas, W.: Networks of Spiking Neurons: The Third Generation of Neural Network Models. *Neural Networks* 10 (1997) 1659–1671.
2. Hill, S.L., Villa, A.E.P.: Dynamic transitions in global network activity influenced by the balance of excitation and inhibition. *Network: Computation in Neural Systems* 8 (1997) 165-184.
3. Abbott, L.F., Nelson, S.B.: Synaptic plasticity: taming the beast. *Nature Neuroscience* 3 (2000) 1178–1183.
4. Bell, C.C., Han, V.Z., Sugawara, Y., Grant, K.: Synaptic plasticity in a cerebellum-like structure depends on temporal order. *Nature* 387 (1997) 278–281.
5. Montgomery, J.M., Madison, D.V.: Discrete synaptic states define a major mechanism of synapse plasticity. *Trends in Neurosciences* 27 (2004) 744-750.
6. Eriksson, J., Torres, O., Mitchell, A., Tucker, G., Lindsay, K., Halliday, D., Rosenberg, J., Moreno, J.M., Villa, A.E.P.: Spiking Neural Networks for Reconfigurable POEtic Tissue. Evolvable Systems: From Biology to hardware. *Lecture Notes in Computer Science* 2606 (2003) 165-173.
7. Torres, O., Eriksson, J., Moreno, J.M., Villa, A.E.P.: Hardware optimization and serial implementation of a novel spiking neuron model for the POEtic tissue. *BioSystems* 76 (2003) 201–208.
8. Moreno, J.M., Thoma, Y., Sanchez, E., Torres, O., Tempesti, G.: Hardware Realization of a Bio-inspired POEtic Tissue. *Proceedings of the NASA/DoD Conference on Evolvable Hardware*. IEEE Computer Society (2004) 237-244.