# IP Generator Tool for Efficient Hardware Acceleration of Self-organizing Maps


**Daniele Giardino, Marco Matta, Marco Re, Francesca Silvestri and Sergio Spanò**

**Abstract** In this paper, authors present an IP generator for FPGA-based hardware acceleration of Kohonen's Self-Organizing Maps (SOM). The IP generator is realized in MATLAB and offers the user the possibility to design an efficient FPGA hardware accelerator with several settings such as the number of features and the number of neurons. The optimization is achieved by applying some approximations to the original SOM algorithm, these modifications do not affect the functionality of the map. The generated IP cores can be used both for training and inference and the software can check the clustering performances.

## 1 Introduction

In recent years, Machine Learning (ML) algorithms were introduced in several fields [1–3]. The increasing interest in ML can be associated both to the high computational capabilities of modern electronic devices and the introduction of new technologies [4–8]. Anyway, ML can be also applied in scenarios such as the Internet of Things (IoT) [9], satellite [10] or unmanned aerial systems (UAS) [11], where intelligence should be moved in specific elements of the network due to the limitations of some devices.

D. Giardino · M. Matta · M. Re · F. Silvestri · S. Spanò (✉)
University of Rome Tor Vergata, Rome, Italy
e-mail: spano@ing.uniroma2.it

D. Giardino
e-mail: giardino@ing.uniroma2.it

M. Matta
e-mail: matta@ing.uniroma2.it

M. Re
e-mail: re@ing.uniroma2.it

F. Silvestri
e-mail: f.silvestri@ing.uniroma2.it



S. Saponara and A. De Gloria (eds.), *Applications in Electronics Pervading Industry, Environment and Society*, Lecture Notes in Electrical Engineering 573, https://doi.org/10.1007/978-3-030-11973-7_59

Currently, the market offers several electronic devices for efficiently implementing ML systems. In this context, FPGAs represent a very interesting choice thanks to their flexibility and high computational power.

The literature proposes multiple examples of FPGA-based hardware accelerators for machine learning. However, the design of such systems requires a considerable effort as it implies the use of hardware description languages (HDL).

In this work, authors present an optimized VHDL code IP generator for Self-Organizing Maps (SOM). The IP generator offers designer the possibility to generate SOM with several settings as, for example, the number of features (dimensions) and the number of neurons. The generated IP cores can be used both for training and for inference stages. Our architecture is able to reduce the hardware complexity of the map while not affecting the clustering performances, this is achieved by some mathematical approximations on the original algorithm.

The SOM algorithm proposed by Teuvo Kohonen [12] uses an unsupervised learning method for mapping high dimensional input data to a low dimensional space that has typically two dimensions. The neurons of a SOM are arranged in a two-dimensional array and, for each of them, an $N$-dimensional weight vector $\vec{m}_i$ is assigned.

In the traditional training mode of SOM, a set of $N$-dimensional input vectors $\vec{x}$, representing the examples for the training process, are fed to the algorithm one at time.

The update process is based on the winner neuron, also known as Best-Matching-Unit (BMU). Since both the weight vectors and the inputs have the same spatial dimensions, they can be represented in an $N$-dimensional space. The winner neuron, identified in this work with a subscript $w$, is the closest one to the considered input.

The update formula for the weight vectors Eq. (1) depends on the considered winner neuron and on a function $h_{wi}$ called *neighbourhood* function. All the neurons are updated simultaneously.

$$\vec{m}_i(t+1) = \vec{m}_i(t) + h_{wi}(t) \cdot \left[\vec{x} - \vec{m}_i\right] \tag{1}$$

The neighbourhood function has its geometric center on the winner neuron and it monotonically decreases as the distance from the point increases. The most common type of neighbourhood function is the Gaussian one (GNF) defined as:

$$h_{wi}(t) = \eta(t) \exp\left(-\frac{\|\vec{m}_i - \vec{m}_w\|^2}{2\sigma^2(t)}\right) \tag{2}$$

where $\eta(t)$ is the learning factor of the network and the variance $\sigma^2(t)$ represents the neighbourhood radius, both should be monotonically decreasing functions.

## 2 Hardware Accelerator Model

In order to efficiently implement SOM on FPGAs, the proposed IP generator introduces some modifications to the original SOM presented in [12]. Equations (1) and (2) are modified replacing the Euclidean distance with the Manhattan distance (less hardware complexity is required) and the Gaussian neighbourhood function is replaced with a base-two exponential function consisting of simple shifts [13].

The proposed Neighbourhood Function updates all the neurons simultaneously in order to get a convergence performance as possible closer to the original Gaussian one. Our neighbourhood function avoids using multiplications, divisions and the computation of any exponential function by converting all the operations into $2^i$ cases. It is well known that multiplications and divisions by power of two can be easily accomplished by simply using arithmetic barrel shifters that are very fast respect to traditional architectures. Considering also the approximation of $e \cong 2$, the Eq. (2) becomes:

$$h_{wi}(t) = 2^{-\left[\frac{\|\vec{m}_i - \vec{m}_w\|}{2^b} + \eta + t_{corr}(t)\right]} \tag{3}$$

The $2^b$ factor is constant and is related to the original neighbourhood radius $\sigma^2(t)$ while $\eta + t_{corr}(t)$ is related to the original learning rate $\eta(t)$. The monotonically decreasing trend of the function has been moved completely to the $t_{corr}(t)$ factor. The $t_{corr}(t)$ trend is designed to decrease by one unit every $2^{t_{bias}}$ epochs where $t_{bias}$ is a constant parameter.

### *2.1 Hardware Architecture*

The implementation of Eqs. (1) and (3) is shown in Fig. 1. Each neuron has its own update structure.

The parameters $m_i$ represent the weight vectors, $m_w$ represents the winner neuron weight vector, $|d_i|$ are the distances between the input $x$ and the i-th neuron, $d_i = \vec{x} - \vec{m}_i$ and $d_m$ is the distance between the winner and the i-th neuron.

## 3 VHDL SOM IP Generator

The IP generator offers to the designers the possibility to configure parameters and to generate the VHDL code using a Graphical User Interface (GUI) realized in MATALB. After the start-up, the program prompts the user for the parameters of the map as shown in Fig. 2. The user can choose the number of features, the number of neurons and the bit size for all the weights. The neurons can be initialized in a hexagonal, grid or random topology covering a certain percentage of the N-d space.

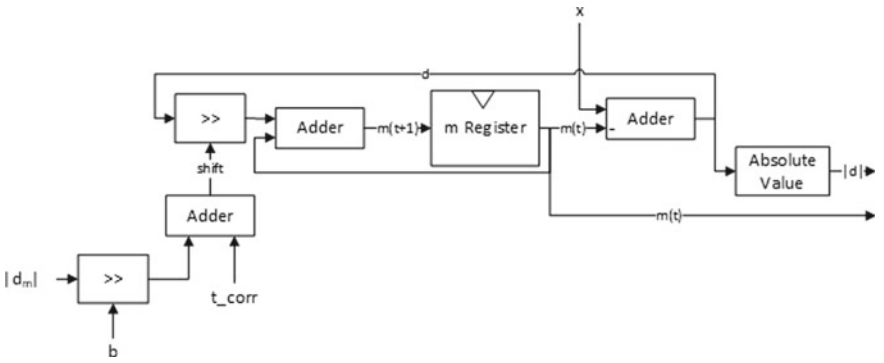


**Fig. 1** Hardware architecture for the update formula of the optimized SOM model

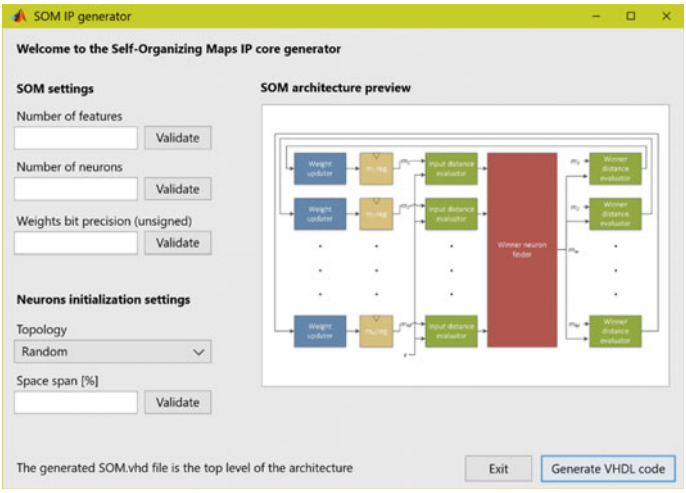


**Fig. 2** Initial configuration prompt

## 4 Simulation

The proposed tool provides a fast way to test the functionality of the generated SOM. The user can train the net with an array of inputs for a certain number of epochs. The software can show the results for a map of maximum 3 features.

As example, Fig. 3 shows the training results of a system where have been used 3 features, 6 noisy clusters (each one consisting of 100 inputs) randomly initialized in a 16 bits quantized space. The map was randomly initialized with 16 neurons, the first plot is the initial state and the second one is the result of the training process. The green smaller dots represent the clusters and the blue circles represent the neurons. In more than 90% of our tests, the clustering performances were satisfying.

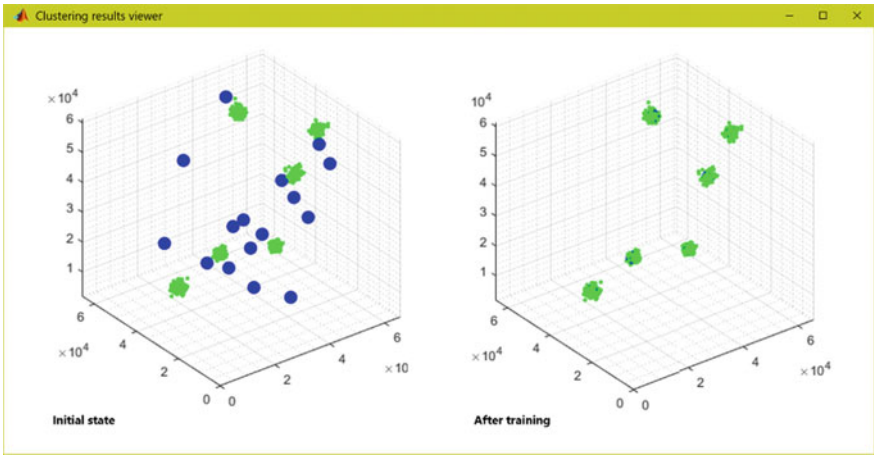


**Fig. 3** Learning simulation results using 3 features

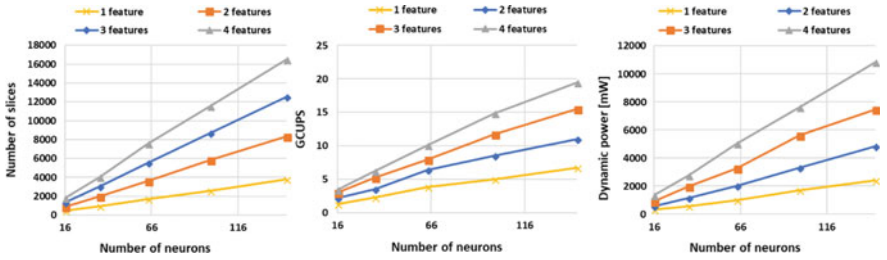


**Fig. 4** Number of slices, dynamic power and GCUPS of different architectures

## 5 Circuit Area, Power and Performances

In order to validate the IP generator, some Synthesis and Place & Route have been performed using the Xilinx Vivado 2017.4 toolchain and the FPGA Virtex 7 xc7vx690t as target device. Experiments have been performed using different SOM configurations. In this section, authors show experimental results for the following configurations:

- 8 bits for representing each weight of the neuron
- 1, to 4 features
- 16, 36, 64, 100 and 144 neurons.

Figure 4 shows the slices, the dynamic power and the Connections Updates Per Second (CUPS) for the various generated architectures. Notice that the power has been estimated using a worst-case approach considering an activity factor of 0.5 on every node of the synthesized network.

The implementation results show how our architectures achieve very high computational performances using a limited amount of hardware resources.

## 6 Conclusion

In this work, we proposed an optimized IP core generator for VHDL code of a hardware accelerator for Self-Organizing Maps. It can accelerate both the learning phase (training) and the recall phase (inference). Thanks to its flexibility, it can be used for any application that requires a high number of neurons or features and low resources, even coupled to a micro-processor [14, 15]. In a future update of the software, we will able to provide an AXI interface that would further facilitate the designer the implementation of our SOM IP core on a System-of-Chip (SoC).

## References

1. Lo Sciuto, G., Susi, G., Cammarata, G., Capizzi, G.: A spiking neural network-based model for anaerobic digestion process. In: IEEE 23rd International Symposium on Power Electronics, Electrical Drives, Automation and Motion (2016)
2. Brusca, S., Capizzi, G., Lo Sciuto, G., Susi, G.: A new design methodology to predict wind farm energy production by means of a spiking neural network based-system. Int. J. Numer. Model. Electron. Netw. Devices Fields **7** (2017)
3. Scarpato, N., Pieroni, A., Di Nunzio, L., Re, M., Salerno, M., Susi, G.: E-health-IoT universe: A review. Int. J. Adv. Sci. Eng. Inf. Technol. **7**(6), 2328–2336 (2017)
4. Cardarilli, G.C., Cristini, A., Di Nunzio, L., Re, M., Salerno, M., Susi, G.: Spiking neural networks based on LIF with latency: simulation and synchronization effects. In: Asilomar Conference on Signals, Systems and Computers (2016)
5. Khanal, G.M., Acciarito, S., Cardarilli, G.C., Chakraborty, A., Di Nunzio, L., Fazzolari, R., Cristini, A., Re, M., Susi, G.: Synaptic behaviour in ZnO-rGO composites thin film memristor. Electron. Lett. **53**(5), 296–298 (2017)
6. Acciarito, S., Cardarilli, G.C., Cristini, A., Di Nunzio, L., Fazzolari, R., Khanal, G.M., Re, M., Susi, G.: Hardware design of LIF with Latency neuron model with memristive STDP synapses. Integr. VLSI J. **59**, 81–89 (2017)
7. Khanal, G.M., Cardarilli, G., Chakraborty, A., Acciarito, S., Mulla, M.Y., Di Nunzio, L., Fazzolari, R., Re, M.: A ZnO-rGO composite thin film discrete memristor. IEEE, ICSE, Article No. 7573608, pp. 129–132 (2016)
8. Acciarito, S., Cristini, A., Di Nunzio, L., Khanal, G.M., Susi, G.: An a VLSI driving circuit for memristor-based STDP. PRIME 2016, Article No. 7519503 (2016)
9. Giuliano, R., Mazzenga, F., Neri, A., Vegni, A.M.: Security access protocols in IoT capillary networks. IEEE Internet Things J. **4**(3), 645–657 (2017)
10. Sacchi C., Rossi T., Menapace M., Granelli F.: Utilization of UWB transmission techniques for broadband satellite connections operating in W-Band. In: 2008 IEEE Globecom Workshops (2008) 1–6
11. Dalmasso I., Galletti I., Giuliano R., Mazzenga F.: WiMAX networks for emergency management based on UAVs. In: IEEE—AESS European Conference on Satellite Telecommunications, pp. 1–6 (2012)
12. Kohonen, T.: The self-organizing map. Neurocomputing **21**, 1–6 (1998)
13. Martín-del-Brío, B., Blasco-Alberto, J.: Hardware-oriented models for VLSI implementation of self-organizing maps. In: International Workshop on Artificial Neural Networks, pp. 712–719 (1995)
14. Cardarilli, G.C., Di Nunzio, L., Fazzolari, R., Re, M., Silvestri, F., Spanò, S.: Energy consumption saving in embedded microprocessors using hardware accelerators. Telkomnika **16**(3), 1019–1026 (2018)

15. Cardarilli, G.C., Di Nunzio, L., Fazzolari, R., Re, M., Lee, R.B.: Integration of butterfly and inverse butterfly nets in embedded processors: effects on power saving. In: Conference Record - Asilomar Conference on Signals, Systems and Computers, Article No. 6489268, pp. 1457–1459 (2012)