Towards the Implementation of a Parallel Hardware Architecture for Spiking Neural Networks

Marco Nuño-Maganda, Miguel Arias-Estrada, Cesar Torres-Huitzil
National Institute for Astrophysics, Optics and Electronics (INAOE)
Luis Enrique Erro No 1, Sta. María Tonantzintla, Puebla, C. P. 72000.
nmaganda@inaoep.mx, ariasm@inaoep.mx, ctorres@inaoep.mx,

Abstract

Artificial Neural Networks are processing models widely explored due to the inherent parallelism. Recently, Spiking Neural Networks (SNNs) have been studied. These models have the advantage of reducing the bandwidth needed for interchanging information among the processing elements, due to the communication scheme based on digital spikes. Many hardware architectures for artificial neural networks have been proposed as an alternative to implementations based on personal computers. In this work, efficient hardware implementation of SNN is addressed. A research overview is presented and some preliminary results as well.

1. Introduction

Artificial Neural Networks (ANNs) are parallel computational models comprised of densely interconnected, simple, adaptive processing units, characterized by an inherent propensity for storing experiential knowledge [1]. ANNs are widely used in a number of applications in which the Neural Networks are usually implemented as a software program on an ordinary digital computer. However, software implementations cannot utilize the essential property of parallelism found in biological Neural Networks.

Spiking neurons differ from traditional connectionist models in the sense that the information is transmitted by means of pulses (or spikes), rather than by average firing rates, allowing spiking neurons to have richer dynamics and to exploit the temporal domain to encode or retrieve information in the exchanged spikes.

2. Motivation

Designing hardware architectures for simulating artificial neural networks is a great challenge because the computational complexity, area greedy, non linear operators and highly dense interconnection shown by these models. The biologically inspired parallelism of ANNs, is lost when ANNs are implemented in modern digital computers as software programs, due to their sequential processing scheme.

Exploring new alternatives with respect to neural networks models is a current research area because recently new neuronal models has been proposed, specifically Spiking Neural Models (SNMs), which represent an alternative to classical models. In [3], the computational power of SNMs has been tested, which is comparable with the classical models, but SNMs requires less hardware resources. SNMs can do the same processing with less processing elements, which is a great advantage when implementing SNN in hardware. It is feasible to explore all the parallelism desired by implementing hardware architectures using Spiking Neuron Models, allowing improving the performance of many SNN applications. For fully exploiting the potential of SNN it would be necessary to develop efficient hardware implementation techniques. In the hardware domain, the challenge is to follow the biological and mathematical trends.

High-performance hardware architectures could be a very interesting research area, due to the high volume of data that demands a high computational capability of neural networks. In [4], a comparison of neural networks implementations in different hardware platform has been made, and only one of the tested hardware platforms has proven to have enough computational power for simulating SNNs. Thus, it is required further investigation to find ways to map efficiently SNN on efficient parallel architectures.

For building high density systems restricted to speed/power consumption, it is necessary to investigate architectural innovations and signal representations to efficiently exploit the abilities of SNN in real world applications where efficient hardware solutions are needed.

3. State of the art

The most interesting spiking neural networks implementations are described below. In [5], a simple model of the neuron is implemented in hardware, but simple neurons lack of representation power and this must be compensated by a higher number of neurons. Adaptation of synaptic weights is implemented with hebbian learning.

In [6], SNNs topologies are explored using genetic algorithms. The neuron model implemented is based in the Leaky-Integrate and Fire model, which exhibits a reduced connectionism schema and low hardware resources requirements. A hardware architecture that used the Dynamic Partial Reconfiguration feature of Xilinx FPGAs is proposed, which allows the reusing of internal logic resources.

In [7], neuron and interconnection architecture in a spiking neural network is given. The spiking neuron model used is LIF (Leaky-Integrate and Fire), where the two basic operations are decay and addition. The proposed architecture can implement three different schemes for adding the weights: parallel processing – serial arithmetic, serial processing – parallel arithmetic and serial processing – serial arithmetic. For the proposed schemes, equations for computing the number of LUTs (Look-Up Tables) required are proposed.

4. Objective

The main objective of this work is to design a high-performance hardware architecture for simulating large spiking neural networks that exploits the parallelism shown in real neurons by implementing a large number of processing elements working in parallel.

5. Methodology

- Review the existing parallelism techniques and conclude which one is the appropriate for implementing spiking neural networks in hardware.
- Define a hardware architecture that allows the user to configure the spiking neural network to adapt it to the problem to be solved. Feed-forward architecture with SpikeProp learning rule is proposed, but other learning algorithms and network architectures are not discarded.
- Define a testbench that allows measuring the results obtained from the proposed architecture and compares them with results obtained from related work.

6. Current State of the Research

In this work, learning algorithms for SNNs have been evaluated. SpikeProp [8] is one of the most explored SNNs learning algorithms, and extensive tests and improvements around this algorithm have been made in the literature. In this work some of these algorithms have been codified in C language, and some improvements have been made to the SpikeProp algorithm. The previous results of our work with this algorithm are presented in table 1.

Algorithms	Inputs	Hidden	Output	Iterations	Accuracy
[8]	50	10	3	1000	97.7
Ours	50	10	3	250	92.0

Table 1. Test of learning algorithms on iris data set

A simplified Leaky-integrate-and fire model has been implemented in Handel-C HDL. The model has an exponential element in its synapses, which is modeled as a weighted first order recursive filter. The output of the synapses is feed to the soma. The threshold value is user definable. The model has two inputs, one for an input spike and one for an 8-bit weight, and one output for the output spike. Synthesis results using a Virtex-II Pro FPGA for a simple Neuron are presented in table 2.

Slice Flip-Flops	12
4 Input LUT	5
Total Slices	8
Total FPGA Slices	13,696
Estimated Neurons in an FPGA (using only an half of total slices)	856

Table 2. Estimated neurons in a Virtex II Pro FPGA

7. References

[1] S. Haykin. Neural Networks: A Comprehensive Foundation. New York: Macmillan College Publishing Company, 1999.

[2] Maass, W. and Bishop, C. M. (1998). Pulsed Neural Networks. MIT Press, 1998.

[3] Simon Johnston, Girijesh Prasad, Liam P. Maguire, T. Martin McGinnity: Comparative Investigation into Classical and Spiking Neuron Implementations on FPGAs. ICANN (1) 2005: 269-274.

[4] A. Jahnke, T. Schoenauer, U. Roth, K. Mohraz and H. Klar, Simulation of Spiking Neural Networks on Different Hardware Platforms, International Conference on Artificial Neural Networks (ICANN), Springer Verlag Berlin, 1997, Page(s): 1187-1192.

[5] Upegui A, Peña-Reyes C, Sanchez E. Hardware Implementation of a Network of Functional Spiking Neurons with Hebbian Learning. BioADIT 2004: 233-243

[6] Upegui A, Peña-Reyes C, Sanchez E. An FPGA platform for on-line topology exploration of spiking neural networks, Microprocessors and Microsystems, Elsevier Science, Volume 29, Issue 5, Pages 211-223, 2005

[7] Schrauwen, B.; D'Haene, M. Compact Digital Hardware Implementations of Spiking Neural Networks. Sixth FirW PhD Symposium. 2005.

[8] S. M. Bohte, H.La Poutré and J.N. Kok. SpikeProp: Error-Backpropagation for in Multi-Layer Networks of Spiking Neurons, 2002, Neurocomputing, November 2002, 48(1-4), pp 17-37