SVITE: A Spike-Based VITE Neuro-Inspired Robot Controller

Fernando Perez-Peña[1,*], Arturo Morgado-Estevez[1], Alejandro Linares-Barranco[2], Manuel Jesus Dominguez-Morales[2], and Angel Jimenez-Fernandez[2]

[1] Applied Robotics Research Lab, University of Cadiz, Spain
{fernandoperez.pena,arturo.morgado}@uca.es
[2] Robotic and Technology of Computers Lab, University of Seville, Spain
{alinares,mdominguez,ajimenez}@atc.us.es

Abstract. This paper presents an implementation of a neuro-inspired algorithm called VITE (Vector Integration To End Point) in FPGA in the spikes domain. VITE aims to generate a non-planned trajectory for reaching tasks in robots. The algorithm has been adapted to work completely in the spike domain under Simulink simulations. The FPGA implementation consists in 4 VITE in parallel for controlling a 4-degree-of-freedom stereo-vision robot. This work represents the main layer of a complex spike-based architecture for robot neuro-inspired reaching tasks in FPGAs. It has been implemented in two Xilinx FPGA families: Virtex-5 and Spartan-6. Resources consumption comparative between both devices is presented. Results obtained for Spartan device could allow controlling complex robotic structures with up to 96 degrees of freedom per FPGA, providing, in parallel, high speed connectivity with other neuromorphic systems sending movement references. An exponential and gamma distribution test over the inter spike interval has been performed to proof the approach to the neural code proposed.

Keywords: Spike systems, Motor control, VITE, Address Event Representation, Neuro-inspired, Poisson, Neuromorphic engineering, Anthropomorphic robots.

1 Introduction

The implementation presented belongs to the Neuromorphic engineer field. The main goal of this discipline is to develop artificial systems which emulate the biological systems. The biological systems, such as: vision and audio systems, speech recognition and control of complex movements, carry out with their tasks with a large efficiency still unknown in the artificial systems. The neuromorphic engineer community use as many features as possible of the human nerve system to reach their goal. Inside this group, engineers try to build up a complete net of neurons in any device.

* This work was supported by the Spanish grant (with support from the European Regional Development Fund) BIOSENSE (TEC2012-37868-C04-02).

One problem to face with is the way of designing and implementing those neuromorphic systems. If we look through the literature, we can find sensors like retinas and cochleas based on VLSI chips [1] and [2] and spike-based control architectures based on FPGA [3], [4] and [5] and also on chips [6]. These neural controllers are based on the third generation of artificial neural networks (ANN): the spiking neural networks (SNN). The main advantages of the FPGA implementations are the reconfigurability and the low cost. In this work we use a FPGA for our purpose and architecture based on blocks which mimic the neural behavior by using spikes to carry out the spatial-temporal information.

Other important issue is about the neural code [7]. From past experiments it is known that neurons do not have the same response to identical stimulus at any time [7]. Thus, they follow a distribution and the most popular one is the Poisson [8], although there are others, like Rate-coded [9], more typical in VLSI or renewal process [8]. The system presented has a deterministic spike source (Rate-coded) which is in front of the neuronal principles. But, due to the way of the translation performed it is achievable a gamma distribution for the inter spike interval at the output of the system. This point has been checked by a comparative with theoretical expected.

Nevertheless, since neurons communicate in a point-to-point manner and it is possible to integrate several thousands of artificial neurons into the same electronic device (VLSI chip or FPGA), new communication strategies has been taken, as the Address-Event-Representation (AER) protocol [10]. AER map each neuron with a fixed address which is transmitted through the interconnected neuron system. By using AER protocol, all neurons are continuously sending information about their excitation level to the central system and it could be processed in real time by a higher layer.

AER is based on the concept which mimics the structure and information coding of the brain. Thus AER let us process the information in real time. That’s one of the reasons of using it: the provided speed. Other one is the scalability that allows it by parallel connections.

The entire architecture consists of an AER retina [1], two layers: processing and actuation one (FPGA implemented) and finally a robotic platform. This paper is focused on the processing layer where is implemented the VITE (Vector Integration To Endpoint) algorithm [11]. The translation into spikes paradigm takes a step forward a complete spike-based processing architecture: from the retina to the robot.

In the next section, the algorithm and its translation are described. Also, details of the blocks are presented. Then in section three we describe both families from Xilinx used; the advantages and disadvantages are enumerated. In section four the results are presented: a comparative between both FPGA models in terms of hardware resources consumption with a brief description of the robotic platform and the test to check the neural code. Finally, to sum up, a discussion about the results achieved is presented.

2 VITE Algorithm

This neuro-inspired algorithm [11] is used for calculating a non-planned trajectory. It computes the difference between the target and the present position. It models planned human arm movements. In contrast to approaches which require the stipulation of the desired individual joint positions, this trajectory generator operates with desired coordinates of the end vector and generates the individual joint driving functions in real-time employing geometric constraints which characterize the manipulator.

In Fig. 1 the block diagram of the algorithm and the translation into spikes domain are shown.

The target position will be supplied by the AER retina [1] to the processing layer (the first one of the architecture). With a mapping function, this layer will generate the spikes according to the reaching target. As was previously noted, this source is deterministic at its firing rate.

Then, the difference between present position and the received from the retina at each time is calculated. The output of the algorithm will be supplied to the second layer of the architecture: the actuation layer. Thus, the FPGA design should include an input and output port in order to carry out with the communication protocol.

Taking a closer look at the translated blocks:

- The Spike Hold & Fire block performed the subtraction between the present position and the target position; both signals are spike streams. The block has two decreasing counters to storage the number of spikes at the input and a combinational circuit to manage these counters and the output.
- The Spikes Integrate & Generate block allows us to integrate the DV (Difference Vector) signal (again a spike stream). This block is composed by a spike counter and a spike generator. The latter uses a parameter called IG_FD (Integrate & Generate_FrequencyDivider) to divide the clock signal and generate the output stream according to this division.
- The Low Pass Filter consists of a Hold & Fire block and an Integrate & Generate block. The output of this second block feed the second input of the hold & fire. The other input comes from previous block in the diagram.

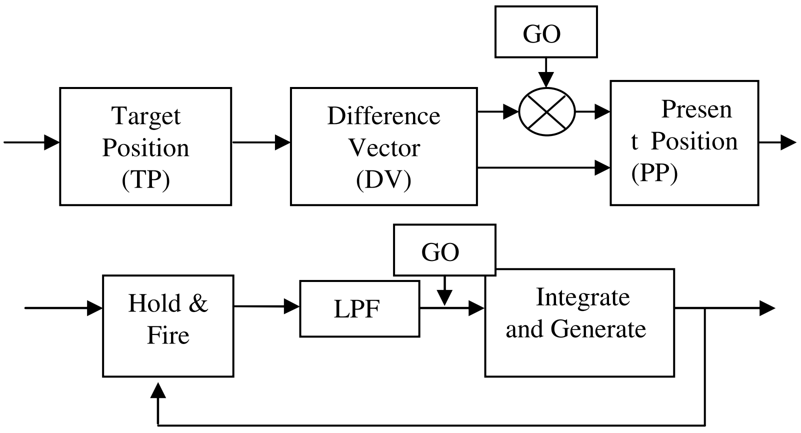

Fig. 1. Top: block diagram of the VITE algorithm. Bottom: block diagram generated from existing spikes processing blocks in [12][13].

- The GO block will be present at the Integrate & Generate block input to put on speed the DV signal. This block copies the task of the multiplier present in the classic algorithm. But if we look through the neuromorphic's engineer point of view, a multiplication is not present in the human nerve system. So, it accomplishes the task injecting spikes according with the desired speed at each time. To do so, it has a counter to generate the number of spikes that should be injected. This block is the key to achieve a Poisson distribution.

These blocks are described in depth in [12] and [13].

At the final design, for each algorithm synthetized in the FPGA a FIFO memory is included to prevent spike loosing problems. The memory receives the spikes produced by the algorithm and delivers to the output interface. So it is very important the total capacity in the device selection.

3 Implementation

Two different FPGA has been used to implement the system using a commercial Xilinx PCB (Virtex5 platform) and the AER-node board (Spartan-6), that has been developed by authors' lab under the Spainsh Research Project VULCANO.

The Virtex-5 device used is the XC5VFX30T which was designed to hold high performance embedded systems. It has two slices per CLB, reaching a total of 5,120 slices and 20,480 flip-flops available. The RAM capacity for this device is up to 2,448 Kb within blocks of 18 Kb. The prototyping board used is AES-V5FXT-EVL30-G from Avnet and it is based on the device described. To achieve the requirements of input/output ports a daughter card was used.

The Spartan-6 device used is the XC6SLX150T which was designed to hold high volume applications at a low cost device. Also it provides high speed serial connectivity. It has two slices per CLB too, reaching a total of 23,038 slices and 184,304 flip-flops available. The RAM capacity for this device is up to 4,824 Kb within blocks of 18 Kb.

The AER Node platform used can be connected in a mesh (using high-speed serial LVDS links) allowing any 2D neuromorphic architecture.

The board has four 2.5Gbps serial ports (SATA connectors) in order to communicate with other neuromorphic chips. These ports take advantage of the eight GTP transceiver ports available in the device.

Furthermore, the board includes two parallel ports of 30-bit to use the standard parallel spike-based AER protocol, both directly and through specialized daughter boards that increase the functionality.

In order to deliver to the FPGA the data necessary for the algorithm execution, a daughter board (plug-in) is connected. It consists of a microcontroller connected to the FPGA through SPI (Serial Peripheral Interface) protocol. The data delivered are configuration parameters for each block. Also the target used as input is delivered.

Figure 2 shows both hardware platforms.

4 Results

We have made two comparisons. On the one hand hardware resources consumption comparative and on the other hand a power consumption comparative between both devices.

4.1 Hardware Resources Consumption

In general, to measure the hardware consumption in a FPGA, two points should be considered: the dedicated resources included to build up complex devices such as multipliers and the configurable logic blocks (CLBs) for general purpose.

The algorithm does not use any complex structure. It just needs counters and hardware to carry out simple arithmetic operations. Therefore the measurements are focused into the available slices at the FPGA.

We have synthesized the algorithm, including the spikes generator and other options like the spikes monitor and the interface with other neuromorphic chips. Table one and two present the data for both devices with the reports obtained.

In these tables, the first column describes implemented elements for each case. The next column shows the amount of slices needed to synthetized the units for each FPGA. The following column represents the maximum number of units that could be allocated for each FPGA. The final column shows device total capacity for all the synthesis performed.

Results evidenced that with additional elements to the algorithm, the amount of slices needed to synthetize is higher. It is remarkable that the interface with other neuromorphic chips does not provoke an increment in the hardware resources consumption. Consequently the final implementation for a complete architecture will consist of the algorithm and the interface. However, the design and test phases need a monitor in order to check the right behavior of the algorithm.

All the results presented in this section, avoid using the FIFO memory because it uses special architecture presents in both devices.

Table 1. Hardware resources consumption details by Virtex5 and Spartan 6 devices

	Number of Slices		Max. blocks in the device		Use by one block (%)	
	Virtex	Spartan	Virtex	Spartan	Virtex	Spartan
Algorithm	208	238	24	96	4.062	1.033
Algorithm plus monitor	478	533	10	43	9.33	2.31
Algorithm plus interface	215	242	23	95	4.2	1.05
Algorithm plus monitor and interface	478	533	10	43	9.33	2.31

4.2 Robotic Platform

The algorithm presented has been applied to a fixed robotic platform to check it. Fig.2 shows the hardware implementation and the result of the position reached when the target is fixed at (123,110) in the frame of reference of the retina.

The robotic platform is a stereo-vision robot with four degrees of freedom powered by DC motors. The power supply requirement of the motors is 24 Vdc. The manufacturer of the motors is Harmonic Drive and the model is RH-8D6006. The structure of the robotic platform is made so that the motors of the y axis are crossed to their axis and have a transmission belt to move the arm.

We propose to use PFM (Pulse Frequency Modulation) to run the motors to take advantage of the spikes produced by the algorithm. Also, PFM is the closest one to the neural system within motor-neurons.

Four units of VITE were replicated in order to control each motor with an independent way. It allows developing synchronized movements by adjusting GO signal in each algorithm.

The main limitation was due to the motor driver and the opto-coupler present in the power stage. These units have a low switching frequency, just 40 Khz and our algorithm generates higher spike rates. Thus, we have modified the spikes generator in the algorithm to generate 40 Kevents per second as it maximum firing rate.

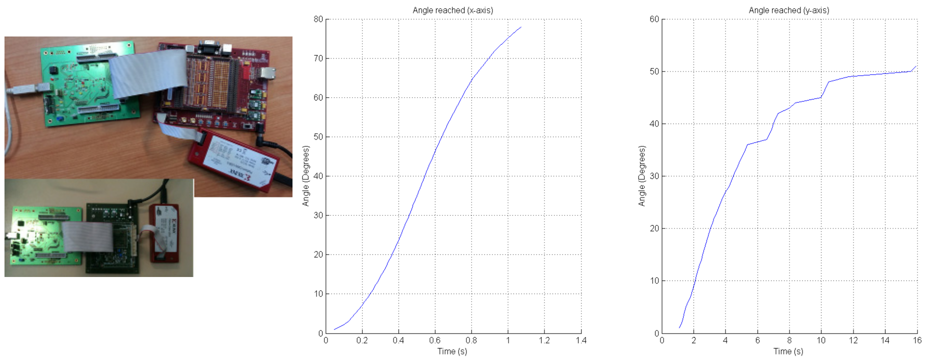

Fig. 2. Left: Virtex 5 at the top and Spartan 6 at the bottom. In both pics appear the programming tool and a special board to monitor the spikes (input and output) [14]. Right: Angle Vs. time reached for both axis with (123,110) input. The retina has 128x128 pixels.

4.3 Inter-Spikes-Intervals Distribution Analysis

In this section, we analyze the InterSpike-Interval(ISI) achieved at the output of the system. As was previously noted, the spike source is deterministic, i.e. it has a one to one mapping between stimulus and response. We are going to compare the inter spike interval with the expected one from a Poisson-like source [15]. The expected ISI of a Poisson process follows an exponential distribution. Also a gamma approach is included in the comparative; it means a renewal process (where the firing probability depends on both: the instantaneous firing rate and the time since the most recent previous spike) [8].

To measure how well the observed distribution of ISIs follows the theoretical exponential or gamma distribution, a comparative between the histogram of ISI read for the speed profile at the input of the integrator superimposed with the theoretical ISI density has been done. As long as the firing rate has dependence with time in our GO block (spikes are injected increasingly within time) and with previous spike in the Hold and Fire block, the ISI follows a gamma distribution.

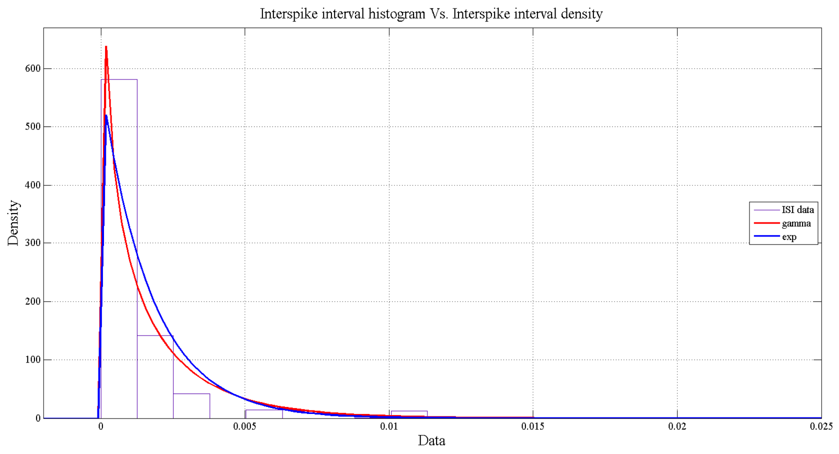

Fig. 3. Comparative between the empiric histogram for ISI and the theoretical defined by exponential and gamma distribution

5 Discussion and Conclusions

We have presented an implementation and a proof of a neuroinspired algorithm in two different devices. The results exhibit a huge advantage using the Spartan 6 device and a well adjust to a Poisson distribution. A total of 96 algorithms can be fitted at the board. The memory requirements are achieved for both devices because we need to storage at maximum 2,048 bytes and both of them have higher capacity. Otherwise, the power consumption is 789 mW for the Virtex-5 device and is 112 mW for the Spartan6 (estimated values with the XPE tool by Xilinx). These values show a big difference between both devices due to the resources used for each one as it has been presented in previous section.

Moreover, the achieved results take into account the slices used by the component in charge of the (Serial Peripheral Interface bus) SPI communication. So, it is possible to avoid that communication and improve the hardware resources consumption.

However, the Virtex device used for the comparison was not the top of the family in contrast with the Spartan 6. With the top Virtex, more algorithms can be fitted but it not allows high speed communication in front of Spartan 6 device.

Also, the proof included to check the neuroinspired features of the algorithm implemented within a digital system shows a good approach to a stochastic gamma distribution for ISI. It reveals a renewal process for the spike train signal even using a deterministic spikes source.

To sum up, the Spartan 6 device can provide a large number of replicated systems in order to control a high number of muscles (mimic by motors) carrying out intended movements in a neuroinspired way. It also allows serial and parallel communication with other neuromorphic chips.

References

1. Lichtsteiner, P., Posch, C., Delbruck, T.: A 128 × 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor. IEEE J. Solid-State Circuits 43, 566–576 (2008)
2. Hafliger, P.: Adaptive WTA with an analog VLSI neuromorphic learning chip. IEEE Trans. Neural Netw. 18, 551–572 (2007)
3. Linares-Barranco, A., Paz-Vicente, et al.: AER neuro-inspired interface to anthropomorphic robotic hand. In: Proceedings of IJCNN, Vancouver, pp. 1497–1504 (2006)
4. Linares-Barranco, A., Gomez-Rodriguez, F., Jimenez-Fernandez, A., Delbruck, T., Lichtensteiner, P.: Using FPGA for visuo-motor control with a silicon retina and a humanoid robot. In: Proceedings of ISCAS 2007, pp. 1192–1195. IEEE Press, New Orleans (2007)
5. Pearson, M.J., Pipe, A.G., Mitchinson, B., Gurney, K., et al.: Implementing Spiking Neural Networks for Real-Time Signal-Processing and Control Applications: A Model-Validated FPGA Approach. IEEE Trans. Neural Networks 18, 1472–1487 (2007)
6. Xiuqing, W., Zeng-Guang, H., Anmin, Z., Min, T., Long, C.: A behavior controller based on spiking neural networks for mobile robots. Neurocomputing 71, 655–666 (2008)
7. Rieke, F., Warland, D., Steveninck, R., Bialek, W.: Spikes Exploring the neural code. MIT Press, Cambridge (1999)
8. Dayan, P., Abbot, L.: Theoretical Neuroscience. MIT Press, Cambridge (2001)
9. Linares-Barranco, A., Jimenez-Moreno, G., Linares-Barranco, B., Civit-Ballcels, A.: On algorithmic rate-coded AER generation. IEEE Transactions on Neural Networks 17(3), 771–788 (2006)
10. Sivilotti, M.: Wiring Considerations in Analog VLSI Systems with Application to Field-Programmable Networks, Ph.D. Thesis, California Institute of Technology, Pasadena CA (1991)
11. Bullock, D., Grossberg, S.: The VITE model: A neural command circuit for generating arm and articulator trajectories. In: Kelso, J.A.S., Mandell, A.J., Shlesinger, M.F. (eds.) Dynamic Patterns in Complex Systems, pp. 305–326. World Scientific Publishers, Singapore (1988)
12. Jimenez-Fernandez, A., Jimenez-Moreno, G., et al.: A Neuro-Inspired Spike-Based PID Motor Controller for Multi-Motor Robots with Low Cost FPGAs. Sensors 12(4), 3831–3856 (2012)
13. Perez-Peña, F., Morgado-Estevez, A., Linares-Barranco, A., et al.: Towards AER VITE: building spike gate signal. In: 19th ICECS, Seville, pp. 881–884 (2012)
14. Berner, R., Delbruck, T., Civit-Balcells, A., et al.: A 5 Meps $100 USB2.0 Address-Event Monitor-Sequencer Interface. In: ISCAS, New Orleans, LA, pp. 2451–2454 (2007)
15. Linares-Barranco, A., Osterb, M., Cascado, D., Jiménez, G., et al.: Inter-spike-intervals analysis of AER Poisson-like generator hardware. Neurocomputing 70, 2692–2700 (2007)