A Low Power Hardware Implementation of Izhikevich Neuron using Stochastic Computing

Aya A. Ismail*[1], Zeinab A. Shaheen*[1], Osama Rashad[1], Khaled N. Salama[2] and Hassan Mostafa[1,3]
[1]*Electronics and Communications Engineering Department, Cairo University, Giza 12613, Egypt*
[2]*King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia*
[3]*Center for Nanoelectronics and Devices, AUC and Zewail City of Science and Technology, Cairo, Egypt*
ayaadelismail@gmail.com, zeinabadel55555@gmail.com, osamarashad90@gmail.com, khaled.salama@kaust.edu.sa and hmostafa@uwaterloo.ca

***Abstract*— This paper introduces the hardware implementation of one of the most popular spiking neuron models which is Izhikevich model. The main target of this implementation is to reduce area and power consumed by the Spiking Neural Network (SNN) neurons as the SNN consists of a large number of neurons to mimic the human brain. Therefore, stochastic computing techniques are used to perform the squaring term that consumes much of the power in the Izhikevich neuron model equations. A hardware implementation of the model is proposed to show the area and power consumption to help the SNN designers to choose between stochastic-based multipliers and the approximate multipliers considering their power, area, and accuracy constraints.**

Keywords*— *spiking neural networks, Izhikevich model, Stochastic computing, ASIC.

I. INTRODUCTION

Spiking neural networks (SNNs) are the third generation of artificial neural networks (ANN). The main concept of these networks is to understand how the brain works and to mimic the natural neural networks. This topic has become an interesting field due to the importance of building systems that can learn to perform tasks without being programmed. Thus, these systems are complementary systems to Von-Neumann systems that reached its peak as there are a lot of computations and tasks that cannot be performed by the traditional Von-Neumann architectures. In addition, even if these computations such as image recognition are performed by using Von-Neumann architectures, they will take much more time than neural systems and consume huge amount of power [1].

There are three types of learning for these systems: Supervised, unsupervised and semi-supervised. In supervised learning, the network is given a set of labelled training data and the algorithm learns to predict the output from the input data such as a set of faces and a set of non-faces and it can then learn to decide whether an image contains a face or not. Whereas in the unsupervised learning, the network is provided by a set of unlabelled data and the algorithm learns to inherent structure from the input data. In the semi-supervised learning, most of the data is unlabelled [1].

In spiking neural networks, neurons communicate by sequence of spikes [1]. When the membrane potential of the neuron reaches a specific value (threshold voltage), the neuron fires a spike and resets its potential. When a neuron fires, it generates a signal which travels to other neuron which, in turn, increase or decrease their potential in accordance to this signal.

One of the trending research topics is neuron modelling which is a mathematical representation of the properties of the neuron and one category of these models is spiking neural networks models [2]. There are a lot of models that vary in their complexity and accuracy.

The main focus of this paper is on the hardware implementation of one of the least-complex and most popular models which is Izhikevich model [3]. The main objective is to reduce the area and power consumed by the squaring term in the Izhikevich model. A lot of approximate techniques have been performed to minimize the hardware, power consumption and area needed to implement this model such as in [4] and [5]. In [6] and [7], piecewise linear models (PWL) are introduced to approximate the quadratic part of the Izhikevich model by crossed lines. This reduces the area and power at the expense of accuracy degradation. CORDIC approximation is introduced in [5] and [8] to approximate this quadratic term.

In this paper, stochastic computing (SC) technique is used to perform the squaring operation. Stochastic computing performs operations using probability instead of arithmetic [9]. A basic feature of SC is that numbers are represented by bit-streams that are processed by simple circuits, whereas the numbers themselves are interpreted as probabilities under both normal and faulty conditions. For example, a bit-stream S containing 25% 1s and 75% 0s denotes the number p = 0.25. P depends on the ratio of 1s to the length of the bit stream, not on their positions so (1,0,0,0), (0,1,0,0), and (0,1,0,0,0,1,0,0) are all possible representations of 0.25 [9]. Other forms of stochastic-based implementations of SNN neuron models are introduced in [10]-[12] where the stochastic features of the memristor device [10] are exploited.

The paper is organized as follows, Section II gives an overview on Izhikevich model, and the stochastic computing techniques. In Section III, the stochastic squaring circuit is introduced to implement the multiplier with accuracy degradation simulations. Section IV presents the hardware implementation results of the proposed SC Izhikevich neuron model. Moreover, a comparison among Stochastic, CORDIC and PWL Izhikevich models [8] is given in Section IV. Finally, a conclusion is drawn in Section V.

* Both authors contributed equally

978-1-5386-8167-1/18/$31.00 ©2018 IEEE

II. Background

A. Izhikevich Model

Izhikevich model is a reduced version of the accurate biologically plausible model Hodgkin-Huxley [3]. It consists of two dimensional system of ordinary differential equations:

$$\begin{cases} \dot{v} = 0.04v^2 + 5v + 140 - u + I \\ \dot{u} = a(bv - u) \end{cases} \quad (1)$$

with the auxiliary after-spike resetting equations:

$$\text{If } v \geq 30\ mV, \text{ then } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \quad (2)$$

where v is the membrane potential and u is the membrane recovery variable. Variables a, b, c and d are dimensionless parameters which determine the spiking type and I represents the synaptic currents or injected dc-currents [3].

B. Stochastic computing

The main importance of SC is that it enables very simple and low-cost implementations of arithmetic operations using standard logic elements. For example, if we have two bit streams at the input of an AND gate and where the probability of finding 1 in the first bit stream is P_1 and the probability of finding 1 in the second bit stream is P_2 then the probability of finding 1 at the output stream is P_1 x P_2 , assuming the two bit streams are independent and uncorrelated, therefore multiplication of two numbers represented in the unipolar representation can be performed by a stochastic circuit consisting only of a single AND gate. Since probability can only take values in the interval [0, 1], stochastic computing is performed only on fractions. Correspondingly, to multiply integer numbers, they should be normalized first and then denormalized after multiplication.

Another motivation to use SC is its immunity to noise as a single bit flip in a long bit-stream will result in a small change in the value of the stochastic number. On the other hand, SC has several problems as the increase in the precision of a stochastic computation requires an exponential increase in the bit-stream length, which results in a corresponding exponential increase in computation time [9].

III. SC Multiplier Design

A. SC Multiplier Design

In this section the design of the SC multiplier used to calculate the term $0.04v^2$ in (1) is presented. Fig.1 shows the multiplier design which was originally proposed by Gupta and Kumaresan [9] with a modification of using two Linear Feedback Shift Registers (LFSRs) for the two Stochastic number generators (SNG) instead of one for better accuracy. Then, a counter is used to convert stochastic number output to binary number.

In this design, the output takes (2^n) cycles and therefore, as n increases, the computational time also increases. However, as n increases, the error decreases and the accuracy is better and accordingly, there is a tradeoff between the computational time and the accuracy. To see the effect of changing the number of bits, a variable word length consisting of two parts (fixed integer bits (10 bits) and variable fractional bits) is adopted and a MATLAB simulation is conducted at different number of fractional bits to show the impact of the variable word length on the accuracy of calculating the squaring term as shown in Fig. 2.

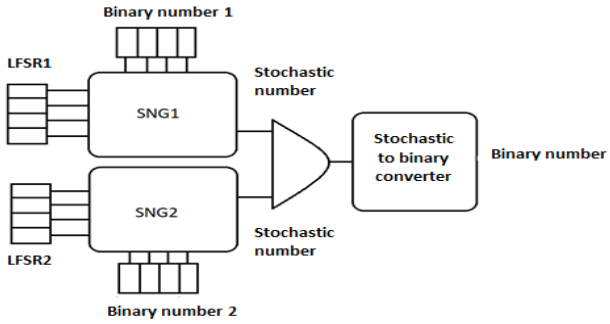

Fig. 1. Stochastic based multiplier design

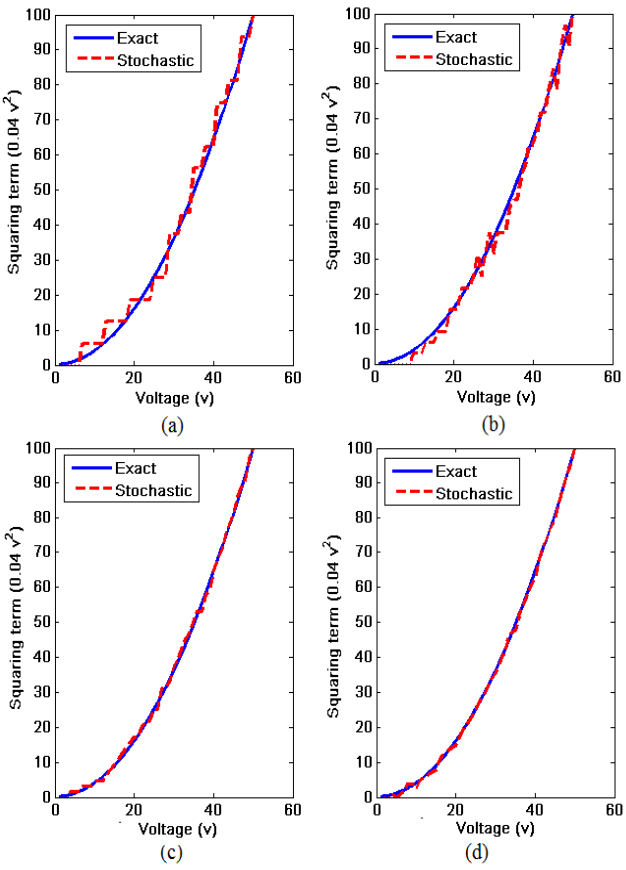

Fig. 2. Output from the stochastic multiplier versus the exact output at different word lengths (n) (a) n=16, (b) n = 17, (c) n=18, and (d) n =19

B. Error Simulation

To see the effect of introducing the SC multiplier on the whole neuron model, the fast spiking response is obtained at different number of bits as shown in Fig.3 and the following model errors, stated in [8], are calculated:

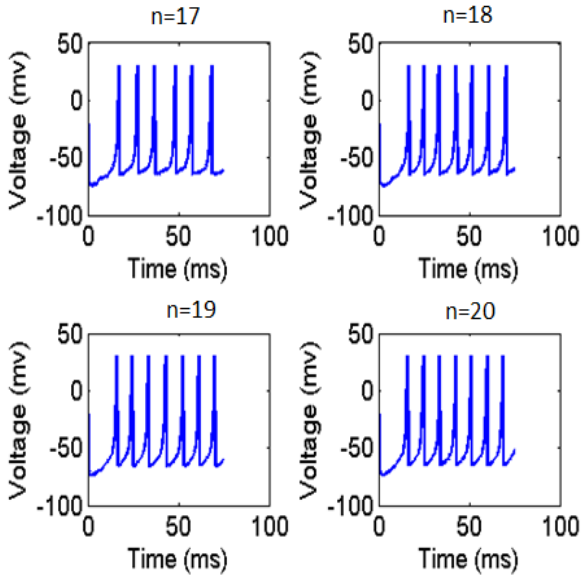

Fig. 3. Fast spiking response at different word lengths

1) Mean Absolute Error (MAE): Calculates the average of the differences between $f(v)$ curves for both the original and the approximated models over the range of v [8].

$$f(v) = 0.04v^2 + 5v + 140 \quad (3)$$

$$MAE = \frac{\sum_i^m |f(v_i)|exact - |f(v_i)|approximate}{m} \quad (4)$$

2) Relative Spike Energy Error (RSEE): Inspired by a common concept in signal processing called "Signal Energy" which is an indication for the resemblance in shape between the spikes generated from both the original and the approximated models [8].

$$RSEE = \frac{\sum_i |v_i^2|exact - \sum_i |v_i^2|approximate}{\sum_i |v_i^2|exact} \times 100 \quad (5)$$

3) Mean Error in Time (MERRt): ERRt measures the time difference between only the first two consecutive spikes. To make the error definition more realistic, ERRt is applied on all the spikes fired in a specific time interval where MERRt is the mean value of ERRt [4], [8].

$$ERRt = \left|\frac{\Delta t_p - \Delta t_o}{\Delta t_o}\right| \times 100 \quad (6)$$

$$MERRt = \frac{1}{m}\sum_i^m ERRt_i \quad (7)$$

Where m is the window of time taken and the error is averaged over it.

IV. Hardware Implementation

Hardware implementation of the proposed design is done using VHDL along with Modelsim 10.4a for simulation and for comparing results against Matlab model. Then ASIC synthesis was performed by using industrial hardware-calibrated TSMC 0.13µm CMOS technology to get the area and power at different number of bits. In addition, a figure of merit (FOM) [7] is defined to show the trade-off among the average error (ERR), area and power and also to find the optimal number of bits to use which is the number at which the FOM is minimum.

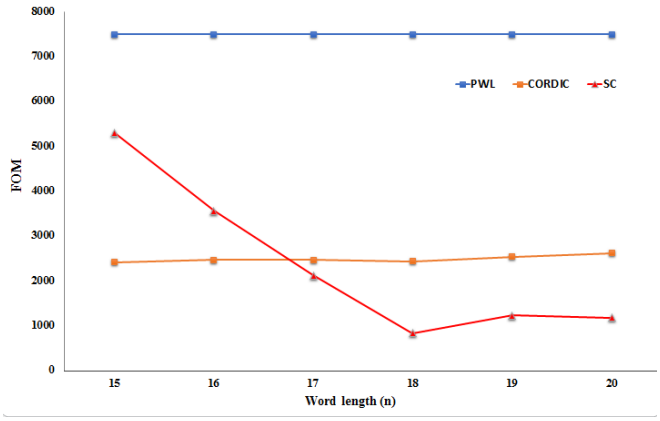

Fig. 4. FOM versus different word lengths

$$FOM = ERR \times power \times area \quad (8)$$

$$ERR = 0.5 \times (MEERt + RSEE) \quad (9)$$

Fig.4. shows the FOM obtained at different word lengths for the SC-based design, the CORDIC-based design and the PWL-based design. It is clear that the optimal number of fraction bits equals 8 (word length n =18) to minimize the FOM. It is also obvious that the FOM in the stochastic-based design is better than the PWL-based design [8] in the whole range of n and better than the CORDIC-based design [8] in the range from n=17 bits to n=20 bits. Correspondingly, all the following simulation results are based on using n = 18.

Table I shows the error results for the different responses. The MEERt error is higher in the mixed mode as the time difference is high between each two consecutive spikes in the proposed and original models as shown in fig.5 (b), while the RSEE is higher in the Tonic spiking response.

TABLE I. MEERT AND RSEE FOR FAST SPIKING, TONIC SPIKING AND MIXED MODE RESPONSES OF STOCHASTIC-BASED IZHIKEVICH MODEL

	MERRt (%)	***RSEE (%)***
Fast spiking	4.82	0.08
Tonic spiking	3.33	7.04
Mixed mode	26.73	0.79
Average error	11.63	2.63

Table II shows a comparison between the results of the hardware implementation of the proposed SC-based design, the original design, the CORDIC-based design and the PWL-based design in terms of power, area and error. The results of the three later designs are introduced in [8]. It is clear that the area of the stochastic-based design is 0.57 the area of the CORDIC design and 0.597 the area of the PWL design, while the power is 0.0813 the power of the CORDIC and 0.089 power of the PWL. On the other hand, the average MEERt error in the stochastic is 8.3 times greater than the CORDIC and 3.8 times greater than the PWL and the average RSEE error in the stochastic is 3.6 times greater than the CORDIC and 1.13 times greater than the PWL.

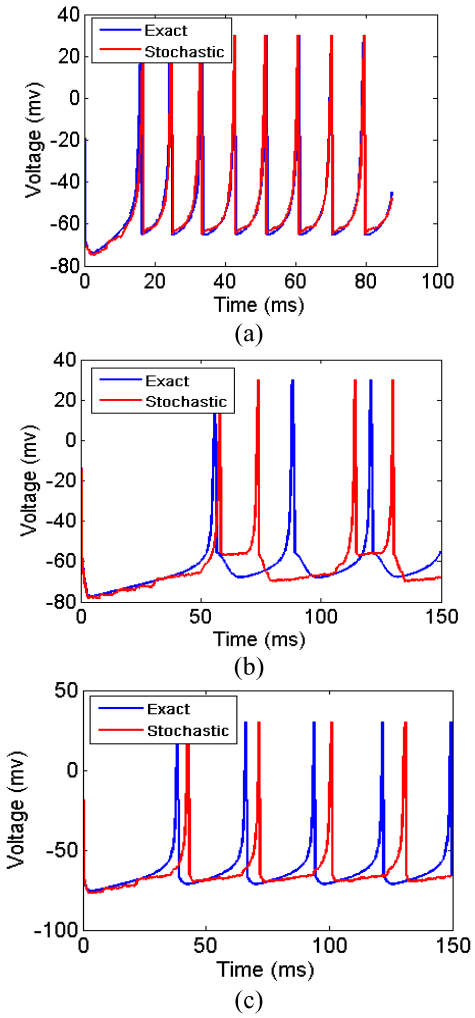

Fig. 5. Original output and output from the Stochastic-based design for, a) Fast spiking response. b) Mixed mode response. c) Tonic spiking responses

V. CONCLUSION

In this paper, the $0.04 * v^2$ term in Izhikevich model has been calculated using stochastic computing technique. Moreover, a hardware implementation and ASIC synthesis have been carried out for different model responses to get the area and power and a comparison is made between the original, Stochastic-based, CORDIC-based and PWL-based Izhikevich models. The Stochastic-based Izhikevich model has shown much less area and power compared to the other models. On the other hand, the error is higher due to correlation between the inputs to the stochastic multiplier. Using stochastic computing has reduced the area by 70.1% and the power by 75.5% at the expense of some accuracy loss.

TABLE II. COMPARISON BETWEEN THE ORIGINAL, STOCHASTIC-BASED, CORDIC-BASED AND PWL-BASED IZHIKEVICH MODELS

	Original	***Stochastic***	***CORDIC***	***PWL***
Area (μm^2)	42059	12584	22088	21076
Power (mW)	0.11	0.0268	0.33	0.3
MEERt (%)	-	11.63	1.39	3.1
RSEE (%)	-	2.63	0.72	2.32
MAE (%)	-	0.4672	0.17	46.85

ACKNOWLEDGMENT

This work was partially funded by ONE Lab at Zewail City of Science and Technology, Egypt and Cairo University, Egypt.

REFERENCES

[1] F. Ponulak and A. Kasinski, "Introduction to spiking neural networks: Information processing, learning and applications," Acta Neurobiol. Exp. (Wars)., vol. 71, no. 4, pp. 409–33, 2011.

[2] A. J. El-Maksoud, Y. O. Elmasry, K. N. Salama, and H. Mostafa, "ASIC oriented comparative analysis of biologically inspired neuron models," IEEE International Midwest Symposium on Circuits and Systems (MWSCAS 2018), Windsor, Ontario, Canada, pp. 504-507, 2018.

[3] E. M. Izhikevich, "Simple model of spiking neurons," IEEE Trans. Neural Networks, vol. 14, no. 6, pp.1569–1572, 2003.

[4] A. Van Schaik, C. Jin, A. McEwan, and T. J. Hamilton, "A log-domain implementation of the Izhikevich neuron model," IEEE International Symposium on Circuits and Systems (ISCAS 2010), pp. 4253–4256, 2010.

[5] M. Heidarpour, A. Ahmadi, and R. Rashidzadeh, "A cordic based digital hardware for adaptive exponential integrate and fire neuron," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 63, no. 11, pp. 1986–1996, 2016.

[6] H. Soleimani, A. Ahmadi and M. Bavandpour, "Biologically inspired spiking neurons: Piecewise linear models and digital implementation," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 59, no. 12, pp. 2991-3004, Dec. 2012.

[7] S. Hassan, H. Mostafa, and K. N. Salama, "An approximate multiplier based hardware implementation of the izhikevich model," IEEE International Midwest Symposium on Circuits and Systems (MWSCAS 2018), Windsor, Ontario, Canada, pp. 492-495, 2018.

[8] A. Elnabawy, H. Abdelmohsen, M. Moustafa, M. Elbediwy, A. Helmy and H. Mostafa, "A low power cordic-based hardware implementation of izhikevich neuron model," IEEE International NEW Circuits and Systems Conference (NEWCAS 2018), Montreal, Quebbec, Canada, pp. 130-133, 2018.

[9] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," ACM Tran. Embedded Computing Systems, vol. 12, no. 2, p. 92, May 2013.

[10] R. Naous, M. Al-Shedivat, E. Neftci, G. Cauwenberghs and K. N. Salama, "Memristor-based neural networks: Synaptic versus neuronal stochasticity," AIP Advances, vol. 6, no. 111304, 2016.

[11] M. Al-Shedivat, R. Naous, G. Cauwenberghs, and K. N. Salama, "Memristors empower spiking neurons with stochasticity," IEEE Journal of Emerging technologies in circuits and systems, vol. 5, no. 2, pp. 242-253, June 2015.

[12] M. Al-Shedivat, R. Naous., E. Neftciy, G. Cauwenberghsy and K. N. Salama, "Inherently stochastic spiking neurons for probabilistic neural computation," 7th International IEEE EMBS Neural Engineering Conference (NER'15), Montpellier, France, April 2015.