Nonlinear Dyn
DOI 10.1007/s11071-017-3584-0

ORIGINAL PAPER

A digital implementation of 2D Hindmarsh–Rose neuron

Moslem Heidarpur · Arash Ahmadi · Nabeeh Kandalaft

Received: 8 July 2016 / Accepted: 22 May 2017

Abstract Different architectures and techniques have developed in the neuromorphic field to mimic and investigate the activity of biological neural networks. This paper presents a set of piece-wise linear approximations of a two-dimensional Hindmarsh–Rose neuron model for digital circuit implementation to achieve higher speeds and lower hardware costs in large-scale implementation of the biological neural networks. The performance of the model was evaluated with a time domain signal error. Synthesis and hardware implementation on a field-programmable gate array, as a proof of concept, indicates that the proposed model reproduces several neuronal behaviors similar to the original model with higher performance and considerably lower implementation costs.

Keywords Hindmarsh–Rose model · Piece-wise linear approximation · Field-programmable gate array (FPGA) · Neuromorphic · Digital implementation

M. Heidarpur
Young Researchers and Elite Club, Kermanshah Branch, Islamic Azad University, Kermanshah, Iran
e-mail: heidarpur.m@gmail.com

A. Ahmadi (✉)
University of Windsor, Windsor, Canada
e-mail: aahmadi@uwindsor.ca

N. Kandalaft
Grand Valley State University, Grand Rapids, MI, USA
e-mail: kandalan@gvsu.edu

1 Introduction

Spiking neural networks (SNNs) are the third generation of neural networks in which neuron models communicate through a sequence of spikes. This is recognized to be a more realistic representation when compared to sigmoidal functions and perceptron neural networks [12]. Due to their similarity to biological neurons, spiking neural networks offer a new approach for studying neural information, processing algorithms in the brain, and finding solutions to applied engineering problems like fast signal processing, event detection, classification, speech recognition, and spatial navigation [34].

Numerous models with different levels of biological information developed to mimic real-world neurons [3,10,16,18,27,37,48]. These models are based on biological neuron processes in the form of differential equations. Some of these models like integrate and fire neuron are too simple and very cheap to simulate and implement even in large scale, but they lack many of the essential features of biological spiking neurons [19]. Models like Hodgkin–Huxley [16] describe the cellular phenomena, physiological parameters, and properties of each components. However, this model is complex and computationally expensive for large-scale simulations and implementations. There is another class of neuron models like Izhikevich [18], AdEx [3], Hindmarsh–Rose [37] that present high-level descriptions with less biological details yet are capable of reproducing many neurocomputational properties of

Published online: 10 June 2017

spiking neurons. These models are more complex than integrate and fire but simpler than HH model for numerical and behavioral studies in the field of information coding and dynamic network analysis.

The three-dimensional Hindmarsh–Rose (3DHR) model was first introduced in 1989 by Rose and Hindmarsh [37]. The model could be recognized as a generalization of Fitzhugh equations [10] or as a simplification of the physiologically realistic model proposed by Hodgkin and Huxley (HH model) [16]. There are many improved version of this model. In [24,25] the model is improved to describe the dynamical behaviors of neuronal activities with electromagnetic induction being considered. Furthermore, autapse-connected neuron model and its biological function and time-delayed feedback on the membrane potential of the neuron is investigated [40]. In 2007, Tsuji et al. [47] proposed a two-dimensional Hindmarsh–Rose (2DHR) model that produced neuronal behaviors similar to the HR neuron model.

There are two major techniques to implement and to simulate spiking neurons networks: computer simulation and very large-scale integration (VLSI) dedicated hardware. Computer-based simulations provide flexible platforms for spiking neural networks where the structure of the network and model parameters (neurons, astrocytes, synapses, etc.) can easily change [29,30,43]. However, as the number of neurons and connections increases, more computational power is needed, which requires supercomputers that are not easily available to many researchers [1,5,44]. VLSI systems consist of analog and digital hardware. They are fast and conduct parallel computations in which many calculations are carried out simultaneously, as in biological spiking networks. Analog systems do not require digital to analog converters to communicate with real-world neurons. They are continuous, fast, and efficient. The disadvantage of these systems is in their analog design complexity, where changing a parameter can lead to an entirely new design. In addition, analog systems are sensitive to noise and process variability [4,17,38,41,49]. Digital systems consume more power, require larger areas, and are slower, but they are more flexible and straightforward to design. Additionally, digital systems are not sensitive to noise and device mismatch which can be problematic in analog systems. Another advantage with respect to the approximation technique used in this work is that discontinuous in the neuron differential equation does not generate problems as in the analog implementation. Digital and analog design techniques are used to implement dedicated hardware for the simulation of SNNs [2]. Furthermore, reconfigurable digital platforms (analog/digital, i.e., FPGAs) provide cheap and widely available platforms which make it possible to implement large-scale networks with high stability and reliability.

Digital platforms are preferred over analog systems because digital logic could be easily minimized by technological improvements, whereas it is not possible to do this in analog circuits due to the thermal noise limitation. The technology node is already at 14 nm today; however, the device used for implementation in this work was a 45-nm device. The implementation using a 14-nm device has the potential to reduce the size of the circuit by approximately four times.

Several digital and analog circuits are proposed in the literature for three-dimensional Hindmarsh–Rose neuron implementations [9,15,21,23,33,36]. In [42], a piece-wise linear approximation was proposed to simplify the 3DHR for lower-cost circuit implementation.

It is important to have simple and effective models while using FPGAs for large spiking neuron networks due to their limited resources. Simpler models like integrate and fire neuron allow implementation of large number neurons on a single chip. However, more complex and detailed model like HR neuron implementation requires more resource utilization that only limited number could be implemented on one FPGA.

Number of neurons and the size of the network can be limited by the available resources such as Slice Registers, Lookup Tables (LUTs), Random Access Memory (RAM) blocks or Digital Signal Processing (DSP) slices. Input/output (I/O) is another resource that can limit the design, yet it can be resolved by using a serial interface or a multiplexed parallel interface.

The implementation of spiking neural networks requires a large number of calculations in order to determine each neuron potential (10 for 2DHR). The number of DSP fast multipliers is limited on FPGAs, and each multiplier degrades the system speed significantly. In general, multipliers are slow and expensive units. Several FPGA implementations are proposed for simulating the spiking neurons networks [7,13,14,26,31,39,46,50].

In this paper, the 2DHR model was selected. The model can mimic all the behaviors exhibited by a real biological neuron, yet it is simpler than the biological HH model. The 2DHR was modified to achieve higher

speeds and lower costs. To avoid the dependence on multipliers, a piece-wise linear model was proposed which can be efficiently implemented on FPGA boards. In addition, a set of piece-wise linear models (PWL) were developed and implemented in large numbers on different digital platforms including FPGAs. Moreover, multiplication was replaced with shift and add operations. To analyze the performance of the proposed model, three error types were calculated in the time domain. The proposed technique can also be implemented on other information processing units in the brain such as astrocytes and synapse models [35,45]. Here the main goal is low-cost implementation of a biological neuron which is capable to be integrated with biological synapse and astrocyte for further investigations on the interactions and effects on STDP [45]. The long-term goal of the research is to find an appropriate hardware as a building block of a neural chip, which consists of biological spiking neurons, biological synapses, and other components of biological neural networks.

The paper is organized as follows: Sect. 2 presents the two-dimensional Hindmarsh–Rose model. Introduction of the PWL technique and the modified models is presented in Sect. 3. The section next explores the accuracy of the proposed model. Section 4 presents the FPGA implementation and results. Finally, Sect. 5 presents the conclusion.

2 Modified two-dimensional Hindmarsh–Rose

The 2DHR model proposed by Tsuji et al. [47] is

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= c\left(x - \frac{x^3}{3} - y + I\right) \\ \frac{dy}{\mathrm{d}t} &= \left(x^2 + dx + a - by\right)/c \end{aligned} \tag{1}$$

The two variables x and y denote the cell membrane potential and a recovery variable, respectively. The "a," "b," "c," and "d" are positive control parameters where "c" represents the timescale. The parameter "I" is the external stimulus or the "membrane current." Because the planar system is too simple to explain the diversity of biological neurons, the auxiliary reset equation is required [8] as:

$$\text{If} \quad x(t) > \theta \quad \begin{cases} x = x_{\mathrm{r}} \\ y = y + y_{\mathrm{r}}. \end{cases} \tag{2}$$

where θ is a threshold, the subscript "r" represents "reset," and x_{r} (resp. $\mathrm{y_r}$) denotes the reset value of x (resp. y). When the spike reaches the threshold value of θ, the membrane potential and the recovery variable will be reset according to Eq. 2.

This model is modified for effective hardware implementation. Two nonlinear (quadratic and cubic) terms in digital implementation require three multipliers. These terms are exchanged with significantly inexpensive operations units such as "absolute value," "addition," and "logic shift" using PWL approximation. The reason for using PWL functions in the physical realization was due to the simplicity of its structure, which is linear in each region of the domain, as well as its higher compactness and evaluation speed [6,11,32].

By replacing quadratic ($f = x^2 + dx + a$) and cubic ($g = x - x^3/3$) terms in the original model with PWL approximations ($\mathrm{f_{pwl}}$, and g_{pwl}) the modified model formulated as:

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= c(g_{\mathrm{pwl}} - y + I) \\ \frac{\mathrm{d}y}{\mathrm{d}t} &= (f_{\mathrm{pwl}} - by)/c \end{aligned} \tag{3}$$

where

$$\begin{aligned} f_{\mathrm{pwl}} = {} & m1\,|x + d/2| + m2(|x + d/2 + m3| \\ & + |x + d/2 - m3|) + m4 \end{aligned} \tag{4}$$

$$\begin{aligned} g_{\mathrm{pwl}} = {} & k1\,|x| + k2\,||x| - k3| + k2(|x| - k3) \\ & + k4\,||x| - k5| + k4(|x| - k5) + k6\,||x| - k7| \\ & + k6(|x| - k7) + k8 \end{aligned} \tag{5}$$

The function "g" is an odd function where "$\mathrm{g_{pwl}}$" only describes the "g" in positive side of X-axis but it satisfies odd symmetry with respect to x. In some of the cases, when considering range of variables and shape of function "f," only the term $x^2 + a$ was approximated with PWL segments, in which case the term "dx" was added to the PWL function.

4-4PWL model In this model nonlinear terms were approximated with 4 linear segments as shown in Fig. 1a,d. Since the 4-4PWL model followed the original model, both in time and in phase domain, it was more efficient to use simpler models.

3-3PWL model In this model, "f" and "g" functions were approximated with three linear segments as shown in Fig. 1b, e. This PWL function can be described by setting zero $m1$, $k6$, and $k7$ variables of Eqs. 4 and 5. This model reproduced all the original model behaviors, so a simpler model was developed and tested.

2-2PWL model In this model, nonlinearities were approximated with PWL approximations and using

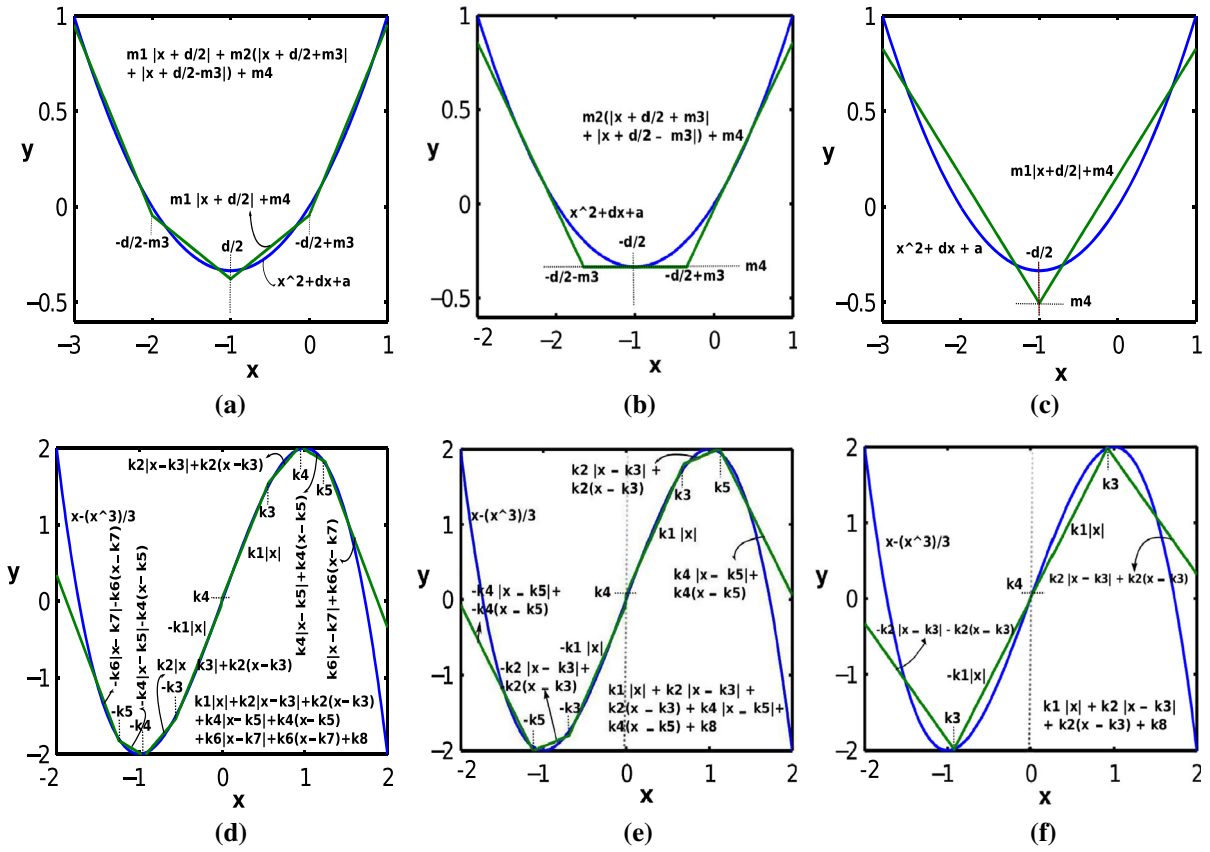

Fig. 1 Piece-wise linear models. **a**, **d** 4-4PWL model. **b**, **e** 3-3PWL model. **c**, **f** 2-2PWL model

only the two segments as shown in Fig. 1c, f. This PWL function can be obtained by setting zero $m2, m3, k4, k5$, $k6$, and $k7$ variables of Eqs. 4 and 5. With proper choice of parameters, this model also was able to emulate the original model firing patterns.

In these models, k and m were defined as constant coefficients that were pre-allocated. Note that the reset condition needed to be added to the above models. The optimum parameter coefficients were found through an exhaustive search algorithm for each neuron type. When these models were compared, the 4-4PWL model had a higher degree of freedom to reach the closest behavior to the original model, but its drawback was that it had a higher implementation cost in comparison with other models.

Figure 2 shows time domain waveforms for the original and PWL models for the various neuron types. As can be seen in the figure, the PWL models exhibit various neuron-like responses and mimic these behaviors as in the original model. Figure 3 compares nullclines of original and proposed PWL models. First row shows nullclines for low value of stimulation where there are two fixed points in PWL models phase planes likewise the HR model. Second row shows responses of models for higher stimulation current where there is only fixed point similar to original model.

3 Proposed model numerical analysis

In this section, it first showed that for a value of stimulation current the response of the original and proposed model is similar and after that this similarity was quantized by calculating errors. However, these errors only showed similarity for a single value of stimulation not necessarily for different stimulation values. "Network behavior" was another method used to show that the output responses of the modified and proposed models were similar for different values of stimulation current. A large number of the similar neurons were stimulated by random values of the stimulation current.

Fig. 2 Simulation of Hindmarsh–Rose and proposed PWL neuron models for different neuronal behaviors

3.1 Time domain errors

To investigate the accuracy of the proposed models, three types of time domain error were examined.

– ERRs: This error was defined as difference of stimulation current for the almost similar firing pattern of original and PWL models. Due to the change in shape of the nullclines in the PWL models, the fixed point of modified models for the same stimulation current may have some differences. The approximation lines could be higher or lower than the original. Therefore, similar behaviors are achieved with higher or lower amounts of stimulation current. This error was calculated as

$$\mathrm{ERRs} = \left| \frac{\Delta I_{\mathrm{pwl}} - \Delta I_{\mathrm{or}}}{\Delta I_{\mathrm{or}}} \right| \times 100 \tag{6}$$

– ERRt: ERRt was defined as the difference in the time interval between the two spikes in the original and PWL models. Initially, the two waveforms were synced so that the apex of the initial spikes matches, and then the time between second and third spikes was used to determine the error (see Fig. 4). In the case of a single spike, the time interval between the stimulation and the apex of the spike was chosen. The corresponding error was therefore formulated as

$$\mathrm{ERRt} = \left| \frac{\Delta t_{\mathrm{pwl}} - \Delta t_{\mathrm{or}}}{\Delta t_{\mathrm{or}}} \right| \times 100 \tag{7}$$
$$\Delta t = t_{\mathrm{apex2}} - t_{\mathrm{apex}}$$

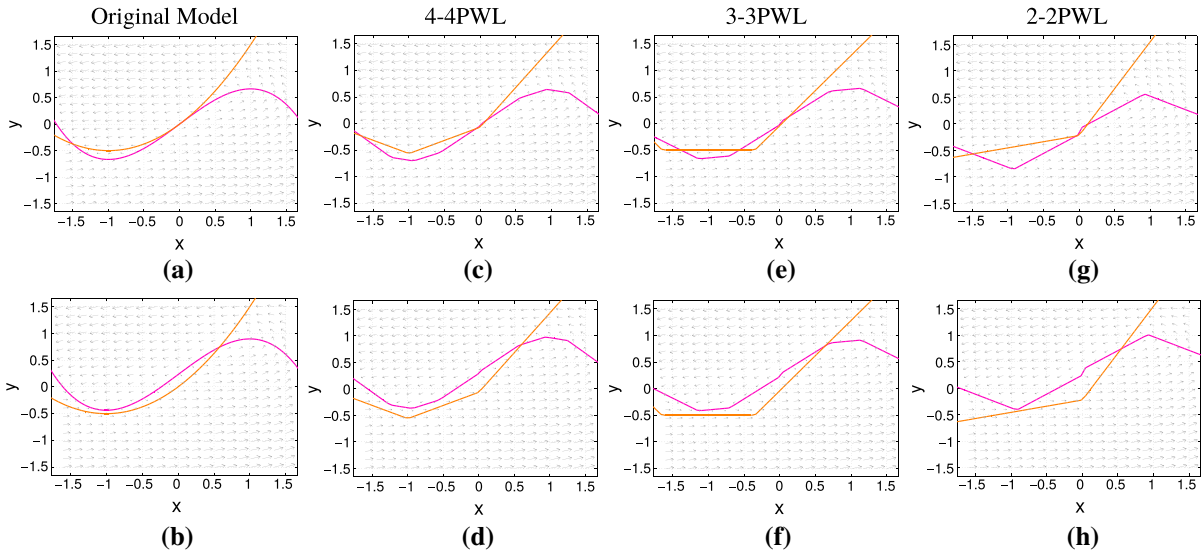

Fig. 3 Nullclines of original model, 4-4PWL, 3-3PWL, and 2-2PWL. Similar to the original model these systems have two interaction points for low injected current; the *second row* shows the state of the models for higher injected current where there is only one fixed point

– NRMSE: The normalized root-mean-square error (NRMSE) was calculated as a difference between the PWL models and the original model. The NRMSE showed lower error values, and the form of the signals showed more similarities. It can be defined as

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(\mathrm{vpwl}(n) - \mathrm{vor}(n))^2}{n}} \tag{8}$$

and normalized RMSE as

$$\mathrm{NRMSE} = \frac{\mathrm{RMSE}}{v_{\max} - v_{\min}} \tag{9}$$

where Vpwl and Vor are the waveforms of the proposed and the original model, respectively. The Vmax and Vmin are maximum and minimum values of the Vor in the calculated NRMSE range. This error was calculated for PWL and original curves (f, f_{pwl}) and (g, g_{pwl}) as $(\mathrm{NRMSE}_f and \mathrm{NRMSE}_g)$. For calculating and quantifying the differences between two single spikes of the modified and the original neuron models, the output waveforms were synced at the apex of a spike and for half of the time interval between the synced apex (t_{apex}) and next apex (t_{apex2}), the NRMSE_s was calculated $([t_{\mathrm{apex}} - (t_{\mathrm{apex2}} - t_{\mathrm{apex}})/2, t_{\mathrm{apex}} + (t_{\mathrm{apex2}} - t_{\mathrm{apex}})/2])$ (see Fig. 4). These errors for different models are presented in their corresponding parameter tables.

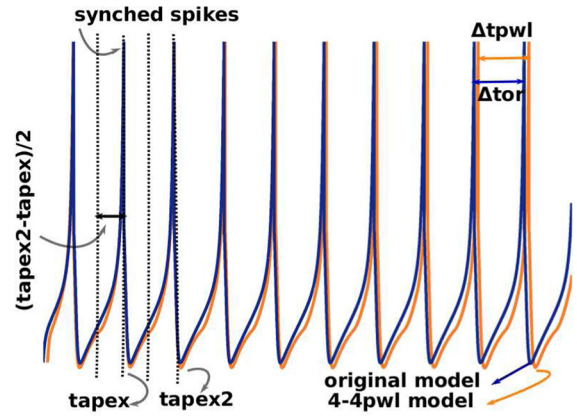

Fig. 4 ERRt error: difference of time interval between two spikes in the original and PWL models

$$(b/3)x^3 + x^2 + (d - b)x + a - bI = 0 \tag{10}$$

and for the modified models:

$$bg_{\mathrm{pwl}} - c^2 f_{\mathrm{pwl}} - Ibc = 0 \tag{11}$$

where f_{pwl} and g_{pwl} are piece-wise linear approximation functions discussed in the previous sections. Solutions for these equations were numerically calculated, and their fixed points were obtained. For determining the stability of equilibrium, the Jacobian matrices and eigenvalues were calculated.

3.2 Network behavior

When analyzing the network behavior of the proposed and original models, it was assumed that 100 synapses were attached to the neuron, with each pre-synaptic neuron firing with a Poisson process of rate $f_{\text{rate}} = 2$ Hz between 200 and 700 ms. The simulation results are shown in Fig. 5 in which the 4-4PWL has the closest behavior to the original model. The most important information from the neural network was the timing of spikes, where their size and shape may not be considered important factors. Therefore, in many applications, it is acceptable to keep the spike timings and ignore the exact form of the membrane potential in the middle of spikes. For instance, in the STDP the spike timing is the most important factor [28].

Therefore, when analyzing the network behavior of the original and proposed models, a network of 2000 randomly coupled neurons was simulated. A schematic spike raster diagram, which consists of a set of spikes over time, is shown in Fig. 5. In this graph, each dot represents a specific neuron spiking at a specific time [22]. The network information is contained in the spike's raster, which shows the spikes of the custom number of neurons in the time axis. Coherent activity can be observed as vertical columns of dots. Even though the network has a randomly connected structure, there is no synaptic plasticity. Inspecting the spike's raster diagram shows that the bursting was synchronized for the active neurons. The neurons were self-organized into groups and show generic rhythmic behavior [20]. Some of the neurons were not synchronized; however, synchronization was dominant in the network as a whole. Figure 6a–d shows the raster diagrams for the original model, 4-4PWL model, 3-3PWL model, and 2-2PWL model. They have differences in details, but, in general, there were no significant differences in the outputs.

4 FPGA implementation

This section presents a hardware design and FPGA implementation results for the modified PWL and original 2DHR models.

4.1 Hardware design

For digital implementation, the 2DHR equations were discretized using Euler method as

$$\begin{aligned} x[n+1] &= x[n]+dt(x[n]-x[n]^3-y[n]+I[n])c \\ y[n+1] &= y[n]+dt(x[n]^2+dx[n]+a-by[n])/c \end{aligned} \tag{12}$$

The discretized equation for the PWL models is shown in Table 2.

In this design multiplication by constant numbers was replaced with "shift" and "add" operations for cost reduction. Therefore, coefficients in the modified model were approximated with 2^n numbers. The values that constants took were limited; they were helpful in order to efficiently implement hardware. The general scheduling diagram for the original and 4-4PWL model is presented in Figs. 7 and 8, respectively. Corresponding to two coupled differential equations, there are two major blocks for calculating x and y. In the scheduling diagrams, the stage which dictates the operation frequency of the clock is

$$\begin{cases} T_{\text{originalmodel}} = T_{\text{Mul}} \\ T_{\text{pwlmodel}} = T_{\text{Add}} \end{cases} \tag{13}$$

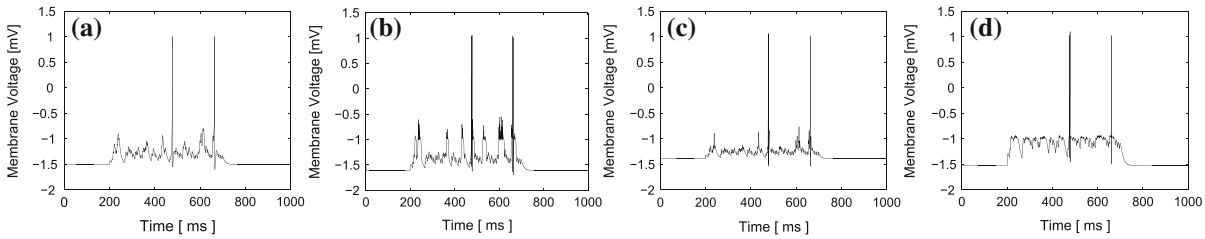

Fig. 5 The neuron model's response to the input of 100 synapses, each one with a pre-synaptic neuron firing with a Poisson process of rate $f_{\text{rate}} = 2$ Hz between time 200 and 700 ms. **a** Original 2DHR model. **b** 4-4PWL model. **c** 3-3PWL. **d** 2-2PWL (note that for closer behavior the weight of PWL models are multiplied by their responsive ERRs error coefficients proportional to the original weights according to Table 1)

Table 1 Calculated errors for PWL models

Models	4-4PWL model					3-3PWL model					2-2PWL model				
	ERRs (%)	ERRt (%)	NRMSEf (%)	NRMSEg (%)	NRMSEs (%)	ERRs (%)	ERRt (%)	NRMSEf (%)	NRMSEg (%)	NRMSEs (%)	ERRs (%)	ERRt (%)	NRMSEf (%)	NRMSEg (%)	NRMSEs (%)
Tonic Spiking	5.2	1.0	2.0	0.7	2.1	16.1	3.5	4.7	1.2	1.5	34.8	13.2	8.6	2.7	3.3
Phasic Spiking	30.0	6.2	7.2	1.96	2.3	130	27.5	8.8	6.7	9.8	−265.0	42.9	9.5	7.8	10.1
Tonic Bursting	7.2	0.8	3.6	2.1	9.8	3.0	1.0	4.8	2.2	14.8	23.8	9.5	4.8	2.2	14.8
Phasic Bursting	140	0.2	2.4	1.5	4.5	150	7.2	9.7	3.4	8.2	−192.0	7.3	10.1	7.2	6.2
Mixed Mode	3.3	2.6	2.4	1.4	2.1	4.3	5.0	6.9	2.3	5.2	13.6	7.9	12.8	4.3	7.4
Spike Frequency Adaptation	1.1	1.1	2.4	1.4	1.6	2.8	10.8	2.5	2.6	4.1	0.5	0.0	8.5	6.8	7.02
Inhibition-Induced Bursting	01.5	02.0	2	1.1	4.4	2.5	6.2	2.6	2.7	4.4	7.7	6.2	3.8	16.1	4.9
Inhibition-Induced Spiking	15.8	21.3	2.3	1.6	3.693	10.0	8.6	3.6	4.3	1.8	30.0	16.8	8.6	5.6	5.1
Integrator	40.2	7.8	3.6	1.3	1.5	2.4	4.2	6.9	1.6	2	3.6	0.3	6.9	1.6	7.9
Spike Latency	32.6	2.2	2.1	0.9	5.0	46.3	9.4	6.2	1.6	6.8	56.8	5.0	7.9	12.7	8.2
Class 1 Excitability	123.8	11.9	0.9	1.6	3.2	−119	17.6	2.5	2.7	3.6	40.6	10.1	6.8	6.9	8.6
Class 2 Excitability	6.7	3.72	1.2	1.2	1.8	18.2	5.0	1.8	2.7	3.7	22.4	12.3	7.2	8.7	9.8
Resonator	22.0	10.1	1.7	3.6	6.1	22.0	3.6	1.9	4.8	7	10.0	5.9	2.3	5.6	5.6

ERRs are defined as difference of stimulation current for the almost similar firing pattern. ERRt measures the difference of time intervals between two spikes in the original and PWL model. NRMSEf and NRMSEg calculate the NRMSE for "f" and "g" functions (Eqs. 4, 5). NRMSEs calculate NRMSE for two spikes of 4-4PWL and original HR model for different firing patterns

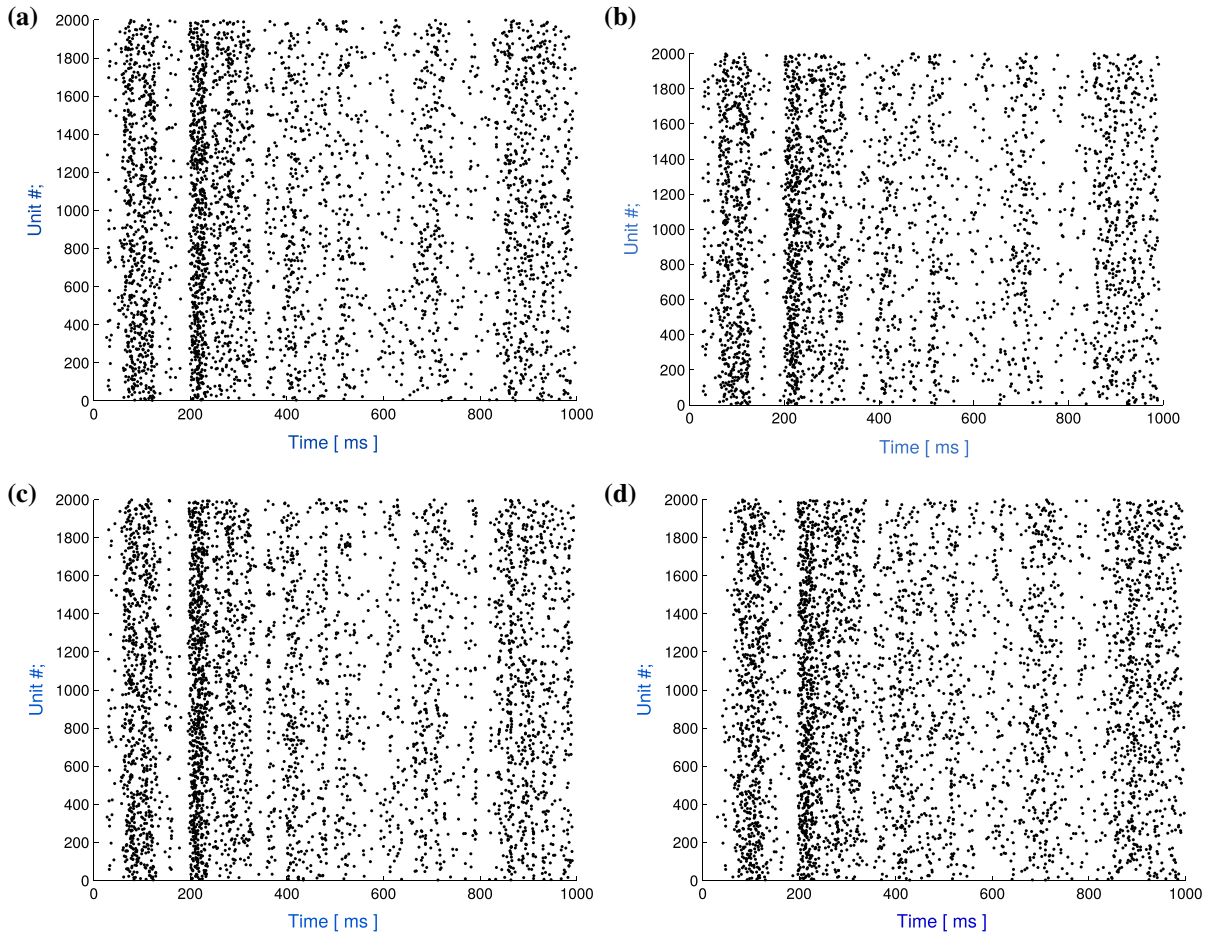

Fig. 6 The spike raster for population of 2000 tonic bursting neurons which are coupled randomly. **a** Original 2DHR model. **b** 4-4PWL model. **c** 3-3PWL. **d** 2-2PWL. (note that for closer behavior the weight of PWL models are multiplied by their responsive ERRs error coefficients proportional to the original weights according to Table 1)

Table 2 Discrete equations describe "x" and "y" for the PWL models

4-4PWL	$x[n+1] = x[n] + dt\,(k1\,\lvert x[n]\rvert + k2\,\lvert\lvert x[n]\rvert - k3\rvert + k4\,\lvert\lvert x[n]\rvert - k5\rvert + k6\,\lvert\lvert x[n]\rvert - k7\rvert + (k2 + k4 + k6)\,\lvert x[n]\rvert + (k8 - k2k3 - k4k5 - k6k7) - y[n] + I[n])$
	$y[n+1] = y[n] + dt\,(m1\,\lvert x[n] + d/2\rvert + m2(\lvert x[n] + d/2 + m3\rvert + \lvert x[n] + d/2 - m3\rvert) + m4 - by[n])$
3-3PWLl	$x[n+1] = x[n] + dt\,(k1\,\lvert x\rvert + k2\,\lvert\lvert x[n]\rvert - k3\rvert + k4\,\lvert\lvert x[n]\rvert - k5\rvert + (k2 + k4)\,\lvert x[n]\rvert + (k6 - k2k3 - k4k5) - y[n] + I[n])$
	$y[n+1] = y[n] + dt\,(m1(\lvert x[n] + d/2 + m2\rvert + \lvert x[n] + d/2 - m2\rvert) + m3 - by[n])$
2-2PWL	$x[n+1] = x[n] + dt\,(k1\,\lvert x[n]\rvert + k2\,\lvert\lvert x[n]\rvert - k3\rvert + k2\,\lvert x[n]\rvert + (k4 - k3) - y[n] + I[n])$
	$y[n+1] = y[n] + dt\,(m1\,\lvert x[n] + d/2\rvert + m2 - by[n])$

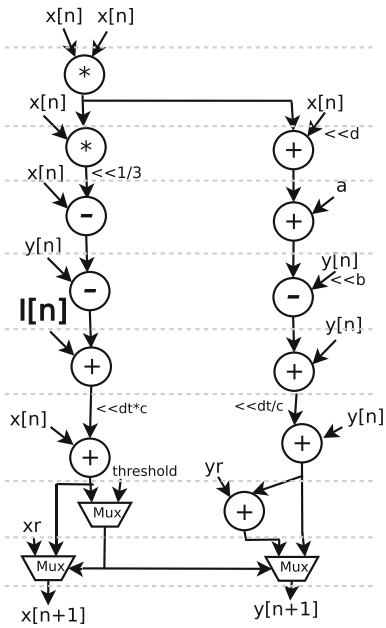

Fig. 7 Scheduling diagram for 2DHR model

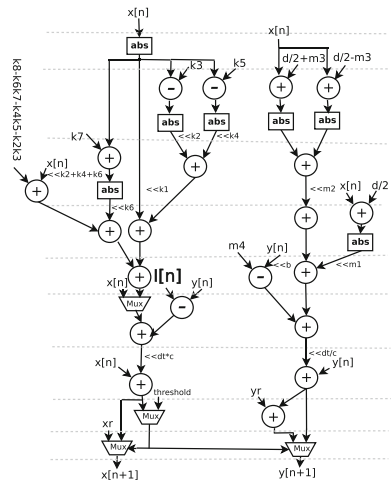

Fig. 8 Scheduling diagram for 4-4PWL model

Table 3 Minimum resources in scheduling of the models

	Original model	4-4PWL	3-3PWL	2-2PWL
Multiplier	1	–	–	–
Adder	2	4	3	2
Multiplexer	2	2	2	2

Considering that T_{Mul} is much larger than a summing operation, it is expected that the PWL models work in higher frequency compared with the original model. The diagram "$\ll$" is used to indicate shift operation. For instance "$\ll k_1$" means to shift the variable by k_1 which k_1 is already written in the form of sum of 2^n numbers like $1/2 + 1/8 + 1/16$. To avoid extra calculations in each step where it was possible, the shift operation was used. In some cases, more steps were added to the scheduling diagrams and, in some cases, more calculations were used. The minimum required resources for each model were different. Excluding the additional calculations required by the shift operation, the minimum resource for the original model and PWL models is shown in Table 3.

Since five bits were needed for the fraction calculation and three bits for the integer calculation, the maximum number of eight calculations can be added to the minimum required resources. Additionally, coefficients were chosen based on the required resources and will be discussed further in the next section.

A floating point typically has better precision, has a higher dynamic range, is easier to develop algorithms, and does not generally encounter issues such as overflow, underflow, and round-off error. In contrast, fixed points have lower precision, have a lower dynamic range, require a constant understanding of the amplitude of the numbers, require knowledge of the accumulation of quantization errors, and require scaling. However, when comparing the cost and performance of these two types of architectures, fixed-point architectures are generally faster and much less expensive than floating point architectures. The main objective of this paper is to reduce the implementation cost; therefore, fixed-point architecture was chosen for the models. This was also the main reason that the PWL method was used since it only requires shift and add operations. Also, as previously mentioned, multiplication of constants was replaced by add and shift operations. If a floating point architecture was used, it was not possible to shift the numbers and multiply each constant without multipliers. Additionally, the cost of using a floating point architecture was not comparable with fixed-point architecture. In the precision point of view, since no multiplication was used (only addition and subtraction) the product may have one more bit width in comparison with the operands (in which multiplication is double). Therefore, only one bit will be lost which does not impose a serious error.

To determine the minimum required bit width, the range of variables, maximum number of shift operations, and minimum precision for the coefficients were taken into account. For the fraction calculation, five bits were dedicated as the least acceptable precision. For calculation of x, the ultimate number of required bits is 16, and the right shift operation is required. Therefore, 21 total bits were selected for the fraction calculation of the variables. The worst possible number of left shift operation is five which also shifts the variable x. Thus, 8 bits are needed for the integer calculation of the variables. To avoid the loss of precision and overflow, the maximum number of 32 bits was used, which is 22 bits for the fraction and 10 bits for the integer.

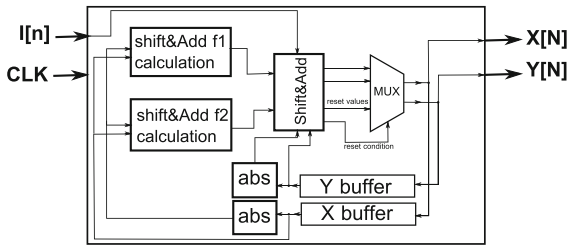

Fig. 9 General overview of the pipelined structure (to avoid the confusing wirings, the clock signal was not drew)

A pipelined architecture for implementation of the PWL models is represented in Fig. 9. Two buffers (X and Y) were utilized to store the output of the x and y modules. These buffers could be used throughout the entire scheduling sequence until new values were received for x and y. In addition, since different stages are required in the pipeline structure of the x and y buffers, some delay units are required for the synchronization of the output variables in order to have proper operation of the structure.

4.2 Implementation results

To prove validation of the proposed PWL models, they were implemented on a XILINX Spartan-6 with XC6SLX9 FPGA development board which utilizes 45-nm process technology. Figure 10 shows photographs of the oscilloscope observing the dynamical behavior of simplest 2-2PWL model for different test cases : Tonic Spiking, Tonic Bursting, Inhibition-Induced Spiking and Bursting, Resonator, Spike Latency, Phasic Spiking, and Phasic Bursting. The stimulation current is applied as in Fig. 2. The device utilization for implementation of the models based on the Hindmarsh–Rose and the proposed PWL

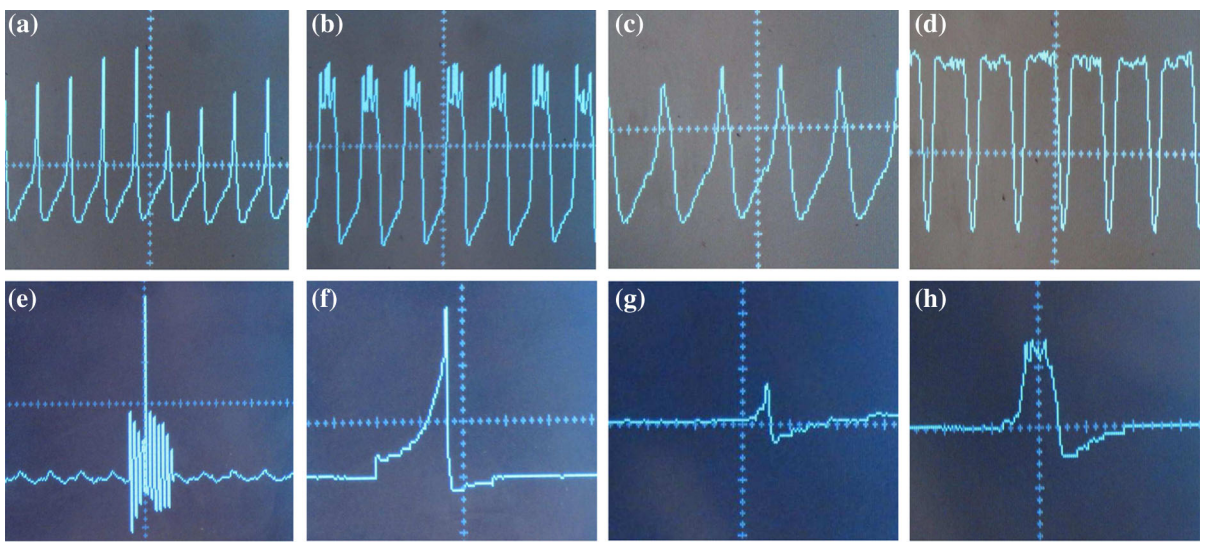

Fig. 10 The 5 bit of the implemented models converted to analog using VGA on board DAC and output signals observed on oscilloscope. **a** Tonic Spiking. **b** Tonic Bursting. **c** Inhibition-Induced Spiking. **d** Inhibition-Induced Bursting. **e** Resonator. **f** Spike Latency. **g** Phasic Spiking. **h** Phasic Bursting (cases: a, b, c, d, f, g :timescale = 100 µs, volt. scale = 500 mv, and case e: time scale = 1 ms, volt. scale = 200 mv)

approximations is summarized in Table 4. The results indicate that hardware implementation of the proposed PWL models was faster compared with the original HR model utilizing general combinational multiplier. This was not surprising since HR model is based on computational expensive operations such as x^3 and x^2 terms, which resulted in a longer critical path in the hardware realization. Using embedded FPGA multipliers resulted in a considerably faster model implementation compared with shift and add algorithm method, however, maximum number of neurons which could be implemented on a FPGA is limited by number of available embedded multipliers on the system. Table 5 compares the number of the neurons that could be implemented in one FPGA for different devices.

Table 6 compares the FPGA device utilization, NRMSD error, and ERRt of the 3-3PWL model implementation with previously published works. Please note that the FPGA devices and synthesizer versions that are used for implementations are different. In addition, these works implement different neuron models and differential equations. Each uses PWL technique to approximate different nonlinearities. For example, Soleimani et al. only approximate one quadratic term of Izhikevich neuron in comparison with Gomar et al. that approximates the exponential term of AdEx neuron. Therefore, the results presented in this table must be considered relatively.

5 Conclusion

In this paper, a set of piece-wise linear approximation models of Hindmarsh–Rose were presented. The simulation and implementation results showed that the proposed modifications follow the same dynamical behavior in time and phase domain as the HR model. The two-dimensional Hindmarsh-Rose neuron model was approximated by piece-wise linear functions. The modified system shows similar dynamical behavior as the original model. These modified models only require arithmetic add and shift operations and were implemented on FPGA without using embedded multipliers. This can limit the number of neurons on a chip. Additionally, the presented models are considerably faster and more space efficient than original model. This implementation approach exhibits different types of neuronal behavior by changing parameter values to reproduce several types of neural behaviors includ-

Table 4 Device utilization and speed of the XILINX Spartan-6 XC6LX9 based on speed optimization goal

Resources	2-2PWL		3-3PWL		4-4PWL		H-R (embedded multiplier)		H-R (shift and add multiplier)		Total Available
	Tonic Spiking	Tonic Bursting	Tonic Spiking	Tonic Bursting	Tonic Spiking	Tonic Bursting	Tonic Spiking	Tonic Bursting	Tonic Spiking	Tonic Bursting	
Slice Registers	544	532	583	578	611	547	422	422	627	627	11,140
Slice LUT's	881	891	1,055	1,185	1,524	1,386	471	471	3,404	3,404	5,720
DSP slices	0	0	0	0	0	0	8	8	0	0	16
Max Speed (MHz)	91.68	89.22	81.78	74.7	73.18	74.5	62.58	62.58	28.41	28.41	250

Table 5 Number of the proposed and original models that can implement on some FPGA devices

FPGA device	2-2PWL	3-3PWL	4-4PWL	Original (DSP mul.)
Spartan-6 XC6LX9	6	5	3	2
Spartan-6 XC6LX100	71	59	41	10
Virtex-5 XC5VTX240T	169	141	98	12
Virtex-6 XC6VLX550T	401	335	225	108

Table 6 Comparison between this and previously published works

Reference	Slice Registers	Slice LUT's	Max Speed (MHz)	DSPs%	NRMSD%	ERRt%	Device
Soleimani et al.	493	617	241.9	0	–	1.54	Virtex-II Pro XC2VP30.
Gomar et al.	388	1279	190	0	4.02	–	Virtex-II Pro XC2VP30
Grassia et al.	646	1048	105	22	-	–	Virtex-5 XC5VLX50
This work	583	1055	82	0	1.5	3.5	Spartan-6 XC6SLX9

ing Tonic Spiking, Tonic Bursting, Inhibition-Induced Bursting, Spike Latency, Class 1 and 2 Excitability.

References

1. Bluebrain|EPFL (2016). http://bluebrain.epfl.ch
2. Benjamin, B.V., Gao, P., McQuinn, E., Choudhary, S., Chandrasekaran, A.R., Bussat, J.M., Alvarez-Icaza, R., Arthur, J.V., Merolla, P.A., Boahen, K.: A mixed-analog-digital multichip system for large-scale neural simulations. Proc. IEEE **102**(5), 699–716 (2014)
3. Brette, R.: Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. J. Neurophysiol. **94**(5), 3637–3642 (2005)
4. Brink, S., Nease, S., Hasler, P., Ramakrishnan, S., Wunderlich, R., Basu, A., Degnan, B.: A learning-enabled neuron array IC based upon transistor channel models of biological phenomena. IEEE Trans. Biomed. Circuits Syst. **7**(1), 71–81 (2013)
5. Cognitive Computation Project (2016). http://ibm.com
6. Cameron, S.H.: Piece-wise linear approximations. Technical Report CSTN-106, Computer Science Division , IIT Research Institute, Chicago, IL (1996)
7. Cassidy, A.S., Georgiou, J., Andreou, A.G.: Design of silicon brains in the nano-cmos era: spiking neurons, learning synapses and neural architecture optimization. Neural Netw. **45**, 4–26 (2013)
8. Chen, S.S., Chrng, C.Y., Lin, Y.R.: Application of a two-dimensional hindmarsh-rose type model for bifurcation analysis. Int. J. Bifurc. Chaos **23**(03), 1350,055 (2013)
9. Dahasert, N., Öztürk, İ., Kiliç, R.: Experimental realizations of the hr neuron model with programmable hardware and synchronization applications. Nonlinear Dyn. **70**(4), 2343–2358 (2012)
10. FitzHugh, R.: Impulses and physiological states in theoretical models of nerve membrane. Biophys. J. **1**, 445–466 (1961)
11. Gallego, G., Berjon, D., Garcia, N.: Optimal polygonal l_1 linearization and fast interpolation of nonlinear systems. IEEE Trans. Circuits Syst. I Regul. Papers **61**(11), 3225–3234 (2014)
12. Ghosh-Dastidar, S., Adeli, H.: Spiking Neural Networks: Third Generation Neural Networks. Springer, Berlin (2009)
13. Gomar, S., Ahmadi, A.: Digital multiplierless implementation of biological adaptive-exponential neuron model. IEEE Trans. Circuits Syst. I Regul. Papers **61**(4), 1206–1219 (2014)
14. Grassia, F., Levi, T., Kohno, T., Saghi, S.: Silicon neuron: digital hardware implementation of the quartic model. Artif Life Robot. **19**(3), 215–219 (2014)
15. Hayati, M., Nouri, M., Abbott, D., Haghiri, S.: Digital multiplierless realization of two-coupled biological hindmarsh; rose neuron model. IEEE Trans. Circuits Syst. II Express Br. **63**(5), 463–467 (2016)
16. Hodgkin, A.L., Huxley, A.F.: A quantitative description of membrane current and its application to conduction and excitation in nerve. Bull. Math. Biol. **52**(1–2), 25–71 (1990)
17. Indiveri, G., Linares-Barranco, B., Hamilton, T.J., van Schaik, A., Etienne-Cummings, R., Delbruck, T., Liu, S.C., Dudek, P., Hafliger, P., Renaud, S., Schemmel, J., Cauwenberghs, G., Arthur, J., Hynna, K., Folowosele, F., Saighi, S., Serrano-Gotarredona, T., Wijekoon, J., Wang, Y., Boahen, K.: Neuromorphic silicon neuron circuits. Front. Neurosci. **5**, 73 (2011)
18. Izhikevich, E.: Simple model of spiking neurons. IEEE Trans. Neural Netw. **14**(6), 1569–1572 (2003)
19. Izhikevich, E.M.: Which model to use for cortical spiking neurons? IEEE Trans. Neural Netw. **15**(5), 1063–1070 (2004)
20. Izhikevich, E.M.: Dynamical Systems in Neuroscience. MIT Press, Cambridge (2007)
21. Kazemi, A., Ahmad, A., Ahmad, S.: A digital synthesis of hindmarsh-rose neuron: a thalamic neuron model of the brain. In: 2014 22nd Iranian Conference on Electrical Engineering (ICEE), pp. 238–241 (2014)

22. Kosslyn, S.M., Andersen, R.A.: Frontiers in Cognitive Neuroscience. MIT Press, Cambridge (1992)
23. Lee, Y.J., Lee, J., Kim, Y.B., Ayers, J., Volkovskii, A., Selverston, A., Abarbanel, H., Rabinovich, M.: Low power real time electronic neuron vlsi design using subthreshold technique. In: Proceedings of the 2004 International Symposium on Circuits and Systems, 2004. ISCAS '04, vol. **4**, pp. 744–747 (2004)
24. Lv, M., Ma, J.: Multiple modes of electrical activities in a new neuron model under electromagnetic radiation. Neurocomputing **205**, 375–381 (2016)
25. Lv, M., Wang, C., Ren, G., Ma, J., Song, X.: Model of electrical activity in a neuron under magnetic flow effect. Nonlinear Dyn. **85**(3), 1479–1490 (2016)
26. Matsubara, T., Torikai, H., Hishiki, T.: A generalized rotate-and-fire digital spiking neuron model and its on-FPGA learning. IEEE Trans. Circuits and Syst. II Express Br. **58**(10), 677–681 (2011)
27. Morris, C., Lecar, H.: Voltage oscillations in the barnacle giant muscle fiber. Biophys. J. **35**(1), 193–213 (1981)
28. Morrison, A., Diesmann, M., Gerstner, W.: Phenomenological models of synaptic plasticity based on spike timing. Biol. Cybern. **98**(6), 459–478 (2008)
29. Nemo (2016). http://nemosim.sourceforge.net/
30. Nest Simulator|The Neural Simulation Tool (2016). http://www.nest-simulator.org/
31. Neil, D., Liu, S.C.: Minitaur, an event-driven fpga-based spiking network accelerator. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **22**(12), 2621–2628 (2014)
32. Oliveri, A., Reimers, M., Storace, M.: Automatic domain partitioning of piecewise-affine simplicial functions implementing model predictive controllers. IEEE Trans. Circuits Syst. II Express Br. **62**(9), 886–890 (2015)
33. Poggi, T., Sciutto, A., Storace, M.: Piecewise linear implementation of nonlinear dynamical systems: from theory to practice. Electron. Lett. **45**(19), 966–967 (2009)
34. Ponulak, F., Kasinski, A.: Introduction to spiking neural networks: information processing, learning and applications. Acta Neurobiol. Exp. **71**(4), 409–433 (2011)
35. Postnov, D., Ryazanova, L., Sosnovtseva, O.: Functional modeling of neural-glial interaction. Biosystems **89**(1–3), 84–91 (2007)
36. Radhika, E., Kumar, S., Kumari, A.: Low power analog VLSI implementation of cortical neuron with threshold modulation. In: 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 561–566 (2015)
37. Rose, R.M., Hindmarsh, J.L.: The assembly of ionic currents in a thalamic neuron I. The three-dimensional model. Proc. R. Soc. B Biol. Sci. **237**(1288), 267–288 (1989)
38. Sharifipoor, O., Ahmadi, A.: An analog implementation of biologically plausible neurons using CCII building blocks. Neural Netw. **36**, 129–135 (2012)
39. Soleimani, H., Ahmadi, A., Bavandpour, M.: Biologically inspired spiking neurons: piecewise linear models and digital implementation. IEEE Trans. Circuits and Syst. I Regul. Papers **59**(12), 2991–3004 (2012)
40. Song, X., Wang, C., Ma, J., Tang, J.: Transition of electric activity of neurons induced by chemical and electric autapses. Sci. China Technol. Sci. **58**(6), 1007–1014 (2015)
41. Starzyk, J.A.: Basawaraj: memristor crossbar architecture for synchronous neural networks. IEEE Trans. Circuits Syst. I Regul. Papers **61**(8), 2390–2401 (2014)
42. Storace, M., Linaro, D., de Lange, E.: The Hindmarsh-Rose neuron model: bifurcation analysis and piecewise-linear approximations. Chaos **18**(3), 033128 (2008)
43. The Brain Spiking Neural Network Simulator (2016). http://briansimulator.org/
44. The Human Brain Project (2016). https://www.humanbrainproject.eu
45. Tewari, S.G., Majumdar, K.K.: A mathematical model of the tripartite synapse: astrocyte-induced synaptic plasticity. J. Biol. Phys. **38**(3), 465–496 (2012)
46. Torikai, H.: Learning of digital spiking neuron and its application potentials. In: Applications of Nonlinear Dynamics: Model and Design of Complex Systems, pp. 273–285. Springer, Heidelberg (2009)
47. Tsuji, S., Ueta, T., Kawakami, H., Fujii, H., Aihara, K.: Bifurcations in two-dimensional Hindmarsh–Rose type mode. Int. J. Bifurc. Chaos **17**(03), 985–998 (2007)
48. Wilson, H.R.: Simplified dynamics of human and mammalian neocortical neurons. J. Theor. Biol. **200**(4), 375–388 (1999)
49. Yamashita, Y., Torikai, H.: A novel PWC spiking neuron model: neuron-like bifurcation scenarios and responses. IEEE Trans. Circuits Syst. I Regul. Papers **59**(11), 2678–2691 (2012)
50. Yildiz, N., Cesur, E., Kayaer, K., Tavsanoglu, V., Alpay, M.: Architecture of a fully pipelined real-time cellular neural network emulator. IEEE Trans. Circuits Syst. I Regul. Pap. **62**(1), 130–138 (2015)