A CORDIC Based Digital Hardware For Adaptive Exponential Integrate and Fire Neuron

Moslem Heidarpour, Arash Ahmadi, *Senior Member, IEEE*, and Rashid Rashidzadeh, *Senior Member, IEEE*

***Abstract*—This paper presents a COordinate Rotation DIgital Computer (CORDIC) based Adaptive Exponential Integrate and Fire (AdEx) neuron for efficient large scale biological neural network implementation. The accuracy of the modified model is investigated by both calculating various errors and bifurcation analysis; both show that the proposed model follows the same signaling, dynamical behavior, and bifurcation pattern as the original model. Network behavior of the original and proposed models are also observed to be very much alike in following the same activity patterns. The model is hardware synthesized and implemented on FPGA as a proof of concept. Measurement results show that the proposed model can reproduce neuronal behaviors similar to the original model. Hardware device utilization and speed also confirm the efficiency of the realized hardware compared with previous works.**

***Index Terms*—Adaptive exponential integrate and fire (AdEx), biological neuron model, CORDIC, digital implementation, neuromorphic, spiking neural network.**

I. Introduction

SPIKING neural networks (SNN) have become an interesting subject for a variety of research fields including computational neuroscience, cognitive computing, artificial intelligence, and neuromorphic. There is a wide range of applications related to this topic such as pattern recognition, data processing, medical diagnoses, autonomous robotics, game playing, etc. [1].

Artificial neural networks are considered basic computational models, which are simplified imitations of biological neurons, replicating basic elements of the nervous system. Based on the in/out signal types, there are three classes of neural networks. The first one represents neurons as perceptrons, which are composed of two parts: sum and threshold. If the sum of weighted inputs reaches the threshold, the output will be one, otherwise it will be zero. This is useful for Boolean function implementation. In the second class, neurons' activation functions are sigmoid. This type also consists of two parts: sum and sigmoid evaluator. In addition to digital functions, this type of networks can approximate analog functions. In these two classes, the normalized firing rate in a period of time, also called the rate coding, determines the output of the network [2]. Although it is considered a strong computational model, there are other processes in the brain, such as classifying visual objects, that are too fast to be explained using only rate coding algorithms [3]. Both experimental and theoretical research suggests that sequence timing of the signals must be considered as another essential parameter of neural coding [4], [5].

The third generation of neural networks uses more biologically meaningful neuron models and incorporates spikes and spatiotemporal data processing schemes [6]–[8]. The neurons of this class have three computational stages: sum of inputs, integrate the sum over time, and a threshold. In this model, when the membrane potential reaches the threshold, the neuron fires a spike and resets its potential. In general, these types of spiking neuron models are mathematically described by ordinary differential equations (ODEs).

Several models with different levels of biological details have been developed to mimic real world neurons [9]–[15]. Unfavorably, as much as the models are biologically detailed, they are computationally expensive and number of floating-point operations (FLOPS) for every step of the simulation increases making large scale simulations costlier. Therefore, choosing a proper model for the simulation of the spiking neural networks is a trade-off between level of biological detail and computational cost.

In this paper, the adaptive exponential integrate and fire (AdEx) has been adopted for the simulation and physical implementation. This model is a hybrid two-dimensional model and undergoes the same bifurcation as other models of this class. Spike initiation is exponential which fits to the experimental results and its parameters could be directly related to biological parameters [16]. This model also shows less unrealistic nonlinearities compared with other two-dimensional neurons [17]. It is also simple enough, and yet able to reproduce almost all of the behaviors exhibited by real biological neurons. This model is compared with some popular models in [18].

Neuron models can be either implemented using VLSI systems or simulated by computer (software) codes. Computer simulations have the advantage of flexibility, full control over model parameters, can be clock driven (synchronous) or event driven (asynchronous), and can simulate multiple neuron models, but even for medium scale networks it falls behind the real time simulation [19]. Since the real power of the brain is believed to stem from massive parallelism, the classic fetch and decode computers do not seem like a very good choice. Many projects have been simulating neural networks on supercomputers [20]–[22] or in some cases software has been

Manuscript received March 1, 2016; revised May 30, 2016, July 19, 2016, and July 24, 2016; accepted July 25, 2016. This paper was recommended by Associate Editor E. Blokhina.

M. Heidarpour is with Young Researchers and Elite Club, Kermanshah Branch, Islamic Azad University, Kermanshah, Iran.

A. Ahmadi is with the Department of Electrical Engineering, Razi University, 67149-67346 Kermanshah, Iran (e-mail: aahmadi70@gmail.com).

R. Rashidzadeh is with the Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON N9B 3P4, Canada.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2016.2598161

developed for PCs [23]–[25], although supercomputers are expensive and are not available for the majority of researchers.

Unlike general purpose architectures, hardware simulators include well developed and dedicated functional units for simulation and mostly work in parallel, which is very desirable in the simulations of SNNs. These types of simulators can be categorized into analog and digital neuromorphic systems. Analog systems, in general, are less power consuming, more area efficient, continuous in time, and therefore have a higher speed, can easily communicate with real world signals and take advantage of the fact that the actual neurons are analog. Having said that, analog implementations generally suffer from some limitations: analog design is not very straightforward and any change in a parameter of the model mostly leads to design changes and might even need a whole system redesign. They are also very sensitive to process variability and suffer from noise. Furthermore, spiking neurons require rather large capacitance, have limited post-synaptic connections due to circuit fan-out constraints, are not sufficiently reliable [19], [26], and mostly rely on digital parts for full functionality [27]. Some of these analog implementations are presented in literature [28]–[35].

Digital systems, on the other hand, are more power consuming, require larger silicon area, discrete in the time, and rather slower; but, they are highly flexible, straightforward to design, can simulate multiple neuron models, not sensitive to process variability and noise, stable, and can benefit from microelectronic technology scalability [19], [26]. The combination of these two techniques have also been used in some projects that uses digital system for the connections and analog neuron components [35]. Recently, field programmable gate arrays (FPGAs) have been used widely for implementing spiking neural networks [36]–[45].

This paper presents a CORDIC implementation of an AdEx neuron model. The main challenge for low cost digital implementation of this neuron model is the exponential function included in the model. In [37], a piece-wise linear exponential (PLE) technique based on the piece-wise linear (PWL) approximation has been used for an efficient hardware implementation of the model. However, the PWL models, unfavorably change the shape of dynamical system nullclines and cause errors such as:

1) For the same stimulat ion current the response of the original and the modified models are different.
2) Because of sharp corners in the segments junction of the PWL model, near the corners a small change in the stimulation current can cause an unrealistic rapid change in the model behavior.
3) The approximation lines can be sometimes above and sometimes below the original curve, so in some intervals by increasing the stimulation current one may get the behavior which is expected by a decreasing current, and vice versa.
4) The range of the system that is approximated with lines is limited to the designer's choice and the PWL model only works on that range. Therefore, for higher or lower parameter values, stimulation of the system might get stuck in a behavior or even show unpredictable responses.
5) The difference in the shape of the spikes may be small, but the accumulated error over time is considerable, and for the same stimulation current at the same time interval the PWL model will fall behind several spikes compared with the original model.

Another alternative method is the look up tables (LUT) realization of the exponential function in which function values is stored in the system memory. The main disadvantages of this method is its high memory usage for high accuracy implementations, which is limited in FPGAs, and can limit the number of neurons that can be implemented in one FPGA. The main drawback of both aforementioned methods is that with the change of the model parameters the whole system has to be redesigned.

The presented CORDIC model, however, is much more accurate compared with previous works and does not come with mentioned errors. The model also has the same performance and resources as the PWL models. The rest of the paper is organized as follows. Section II introduces the AdEx neuron model, the CORDIC based model, and the simulation of both models in the time domain. Section III investigates the accuracy of the model in comparison with the original one in three aspects: error analysis, dynamical analysis, and network behavior. The next section presents details of the hardware architecture and implementation of the CORDIC AdEx neuron. The first subsection proposes an architecture for the model implementation, where the second subsection describes the architecture using hardware description language (HDL, Verilog). Furthermore, implementation of the model on FPGA and corresponding experimental measurements are presented as a proof of concept. Section IV discusses the results and compares them with the previous published papers, and finally, the last section concludes the paper.

II. Neuron Model

A. AdEx Model

AdEx is a two-dimensional model which can be considered as an improved form of the integrate and fire neuron model [46]. The model describes a neuron by a system of nonlinear differential equations with an auxiliary reset equation as [18]

$$C\frac{dV_1}{dt} = -g_L\Delta T\exp\left(\frac{V_1 - V_T}{\Delta T}\right) + I_{\rm app} - w_1 \quad (1)$$

$$\tau_w\frac{dw_1}{dt} = \alpha(V_1 - E_l) - w_1 \quad (2)$$

reset equation

$$\text{if} \quad V_1 > 0 \ \text{ then } \begin{cases} V_1 \to v_r \\ w_1 \to w_r = w_1 + b. \end{cases} \quad (3)$$

The coupled differential equations of (1) and (2) describe the membrane potential V_1 and the adaptation current W_1, where $I_{\rm app}$ represents the applied stimulation current. The difference between the model in [46] and AdEx is that in the AdEx model, the upswing of the action potential is described by an exponential term, which is more similar to biological neuron behavior.

When the action potential reaches the threshold, V_t, the exponential term causes a rapid rise of the membrane potential; once it reaches the peak, which in the above equation is zero, it will be reset to V_r (3). Other parameters of (1) and (2) are:

- C: Total membrane capacitance (pF);
- g_l: Total leak conductance (nS);
- E_l: Effective rest potential (mV);
- ΔT: Threshold slop factor (mV);
- V_T: Effective threshold potential (mV);
- V_r: Resetting potential (mV);
- τ_w: Adaptation time constant (ms);
- a: Subthreshold adaptation (nS);
- b: Spike-triggered adaptation (pA);
- $I_{\rm app}$: Applied current (pA).

With proper choice of the parameters, this model is able to reproduce many physiological firing patterns such as tonic spiking, adaptation, initial or regular bursting, transient spiking, irregular spiking, etc. For more details please refer to [14], [18], and [47].

B. CORDIC Based AdEx

This section presents a modified AdEx model, which is suitable for low-cost and large scale hardware implementations. The factor that prevalently motivates us to utilize the CORDIC algorithm is the simplicity of its structure that only requires add and shift operations, which can be effectively implemented on digital platforms.

The coordinate rotation digital computer is a class of the hardware-efficient algorithms for trigonometric, hyperbolic, and logarithmic functions. The CORDIC algorithm was first introduced in [48] and [49]. This algorithm is generalized for the calculation of hyperbolic and exponential functions, multiplications, divisions, and square roots. CORDIC is generally faster than other approaches when a hardware multiplier is not available, such as low-cost microcontrollers and FPGAs, or when there are restrictions in the number of available gates to implement the function. The algorithm can be written in the general form

$$\begin{aligned} x_{i+1} &= x_i - m.\mu_i.y_i.\delta_{m,i} \\ y_{i+1} &= y_i + \mu_i.x_i.\delta_{m,i} \\ z_{i+1} &= z_i - \mu_i.\alpha_{m,i} \end{aligned} \tag{4}$$

which can be described as a rotation of the plane vector $v_i = (x_i, y_i)^T$ to $v_{i+1} = (x_{i+1}, y_{i+1})^T$ at each iteration. Here $m \in \{1, 0, -1\}$ specifies a circular, linear, or hyperbolic coordinate system, respectively, and μ_i shows the rotation direction and δ is

$$\delta_i = d^{-s_{m,i}} \tag{5}$$

where $s_{m,i}$ is an integer number and d is the employed radix, for instance, $d = 2$ represents a radix-2 number system, $\alpha_{m,i}$ is the rotation angle, and z_i keeps track of the rotation angle. In the CORDIC algorithm, logarithmic and exponential functions are derived from hyperbolic computations, the rotation and vectoring mode are discussed in [50]. Accordingly, for hyperbolic functions, equation (4) can be written as

```
1  //assign initial values
2  z=fraction(x);poweroftwo=0.5;
3  expx=1;
4  //pre-calculated a elements
5  a=[exp((1/2)*(1:n))]
6  //Determine the weights and
7  //calculate products
8   for i from 0 to n do{
9    if ( poweroftwo < z ){
10     z=z-poweroftwo;
11     expx = expx * a(i);
12    }
13   poweroftwo=poweroftwo/2;
14    }
15 //Account for factor EXP(int(x))
16 for i = 1 : int(x){
17   if ( x < 0 ){
18     expx=expx/e;
19     else
20     expx=expx*e;
21   }}
```

Fig. 1. The pseudocode of search algorithm.

$$\begin{aligned} x_{i+1} &= x_i + \mu_i.y_i.2^i \\ y_{i+1} &= y_i + \mu_i.x_i.2^i \\ z_{i+1} &= z_i - \mu_i.\alpha_i \end{aligned} \tag{6}$$

where $\alpha_i = \operatorname{arctanh}(2^{-i})$ are microrotations in radians and

$$\mu_i = -1 \text{ if } z_i < 0 \text{ and } \mu_i = +1 \text{ otherwise.} \tag{7}$$

In our presented approach the exponential term in (1) is realized using the CORDIC algorithm. Corresponding pseudocode is presented in Fig. 1. This code calculates e^x, where the input is variable x and the output is stored in variable $\exp x$. It uses the well-known fact that

$$\begin{aligned} e^{(x+y)} &= e^x \times e^y \\ e^{(x \times y)} &= (e^x)^y \end{aligned} \tag{8}$$

where the input number, x, is first decomposed into integer and fractional parts $x = x_{\rm int} + z$. The table of factors, is in its binary representation for the fractional part as

$$z = \frac{z[1]}{2} + \frac{z[2]}{4} + \frac{z[3]}{8} + \cdots \tag{9}$$

where $z[i]$ is either 0 or 1. The first loop then computes

$$\left(e^{\left(\frac{1}{2}\right)}\right)^{z[1]} \times \left(e^{\left(\frac{1}{4}\right)}\right)^{z[2]} \times \left(e^{\left(\frac{1}{8}\right)}\right)^{z[3]} \times \cdots \tag{10}$$

where the second exponentiation is to read as

$$(z[i] == 1) \quad ? \quad \exp\left((1/2)^i\right) : 1 \tag{11}$$

that is, only factors with $z[i] == 1$ are actually presented in the product.

The pre-calculated values of $e^{(1/2)}, e^{(1/4)}, \ldots, e^{(1/n)}$ are already stored in an array in line 5. The **FOR** loop in line 8 calculates the exponential function value for the fraction part of

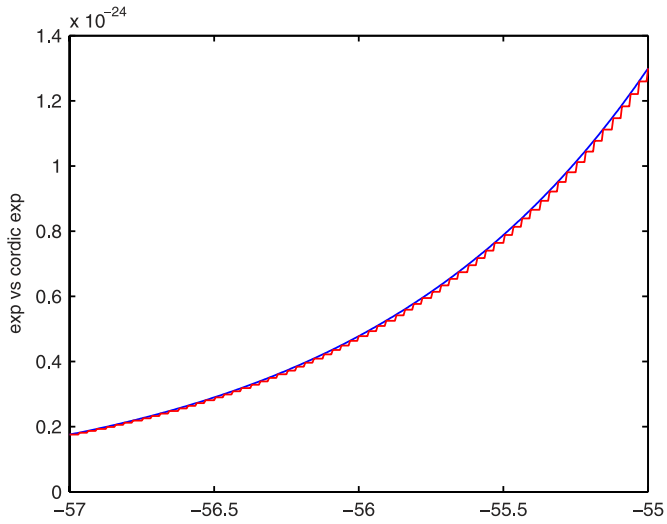


Fig. 2. Comparison between computer simulation and CORDIC implementation result of the exponential function. Functions are plotted for a very short time interval to highlight their differences. The blue and red lines show the exponential and the CORDIC exponential, respectively. For the larger intervals two lines almost become one, and the difference is not visible.

the x with e^{-n} precision. Selecting n, depends on the function minimum precision requirement. In this case, by some trial and error, "n" was found to be 4. The exponential function value of x_{int} is calculated in the second **FOR** loop in line 16

$$e^{(1)} \times e^{(2)} \times e^{(3)} \times \cdots \times e^{(x_{\text{int}})} \tag{12}$$

or

$$e^{(-1)} \times e^{(-2)} \times e^{(-3)} \times \cdots \times e^{(-x_{\text{int}})} \tag{13}$$

depending on the sign of x. The multiplication in constants was also replaced with add and shift operations.

C. Simulation

Fig. 2 shows the exponential function calculated by a floating point software simulation vs. the CORDIC method realization for $n = 4$. The functions are evaluated on a short interval, because for larger intervals two lines become so close that the difference between them is not visible. As indicated by the graphs, the CORDIC exponential function follows the original one with very small and negligible error. An error analysis is presented in the next section. Numerical (floating-point) computer simulations of the original and the proposed model are shown in Fig. 4. As it can be seen, there is no distinctive difference between the proposed and the original models. But for a quantitative investigation of the proposed model accuracy, two types of error calculations are presented in next section.

III. Accuracy Investigation

A. Error Analysis

1) ERRt: It is notable that any difference in signal shaping between the original and the modified models can cause considerable differences between spikes' timing and/or intervals in long spike trains. In other words, the modified model

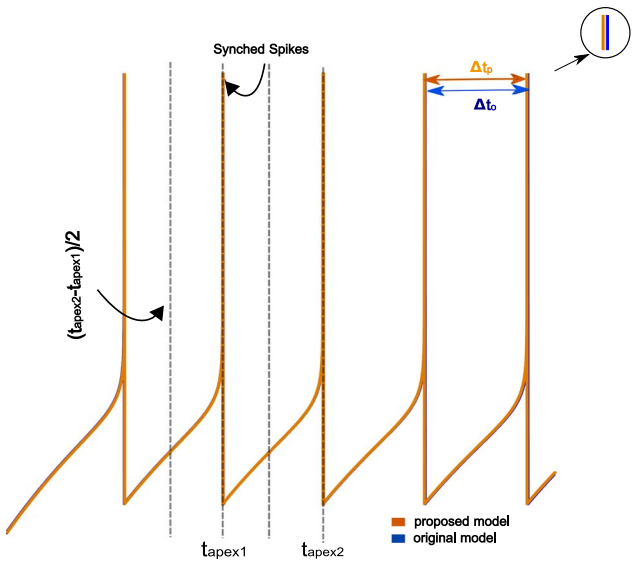


Fig. 3. ERRt error: The difference of the time interval between two spikes in the original and PWL models. The result is obtained from a computer simulation of the original and the proposed models.

TABLE I
ERRt and NRMSD for Different Neuron Behaviors

Error Type	Tonic Spiking	Initial Burst	Regular Bursting	Delayed Accelerating
Errt	%0.3851	%0.1562	%0.1351	%0.1592
NRMSE	%0.0399	%0.9403	%0.6543	%0.1013

may have a time lag in comparison with the original one. ERRt is defined as the time interval differences between two spikes in the original and the proposed model. First, two waveforms are synchronized in the apex of a spike, and then the time between the synced and the next spike is considered for the calculation of this error (see Fig. 3). The error is formulated as

$$ERRt = \left| \frac{\Delta t_p - \Delta t_o}{\Delta t_o} \right| \times 100$$
$$\Delta t = t_{\text{apex2}} - t_{\text{tapex1}} \tag{14}$$

where Δt_p, and Δt_o are the time intervals between two spikes of the proposed and the original models, respectively.

2) NRMSD: The normalized root mean square deviation (NRMSD) is used as the difference between the proposed and the original models [51]. Shaping of the signals are more similar when the error value is lower. This error is mathematically defined as

$$RMSD = \sqrt{\frac{\sum_{i=1}^{n} (vp(n) - vo(n))^2}{n}} \tag{15}$$

and normalized RMSD as

$$NRMSD = \frac{RMSD}{v_{\max} - v_{\min}} \tag{16}$$

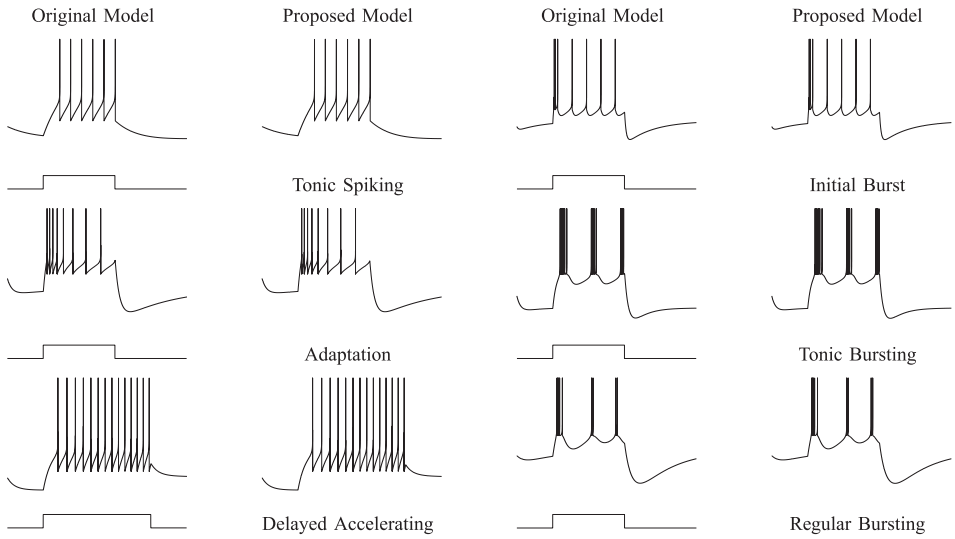


Fig. 4. Computer simulation of the original and modified models for different neuronal behaviors. The stimulation currents are shown below the corresponding neuron spikes.

where V_p and V_o are the waveforms of the proposed and the original model, respectively. $V_{\max}$ and $V_{\min}$ are the maximum and minimum values of V_o in the calculated NRMSE range. For the curves in the Fig. 2 the RMSD and NRMSD are calculated over the range of $[-80,\ 0]$ as 5.34×10^{-4} and 1.45×10^{-3}, respectively. Furthermore, to have a quantitative similarity comparison between the spike signals of the modified and original neuron models, the output waveforms are synchronized in the apex of a spike then the NRMSE is calculated for half of time interval between the synced apex (t_{apex}) and next apex (t_{apex2}), (in other words: $t_{\text{apex}} - (t_{\text{apex2}})/2 < t_{\text{apex}} < t_{\text{apex}} + (t_{\text{apex2}})/2$), see Fig. 3. Different error analyses results for different neuron behaviors are shown in Table I. As indicated by the table, the error values are very low and the waveform of the models are very closely matched. Please note that these errors are calculated for specific stimulation current values, which does not necessarily prove the validation for other stimulation values. Therefore, in the next two subsections the accuracy of the proposed model is investigated for a range of random stimulation current values.

B. Dynamical Analysis

In this section, the original and modified models are briefly analyzed from a dynamical system point of view. For this purpose, the number of equilibrium points and their stability are discussed. The equilibrium points are intersections of nullclines. If the initial point is near equilibrium, then the solutions may converge to or diverge from the equilibrium for stable or unstable equilibrium, respectively [52]. The nullclines of the AdEx neuron can be calculated by [53]

$$\frac{dv}{dvt} = 0 \quad \text{and} \quad \frac{dw}{dvt} = 0 \tag{17}$$

that gives

$$\begin{aligned} w_v &= -g_L(v - E_L) + g_L \Delta t \exp((v - v_T)/\Delta t) + I \\ w_w &= a(v - E_L) \end{aligned} \tag{18}$$

With a different set of the parameters, diverse neuronal behaviors can take place through different bifurcation scenarios. One of the general bifurcations [saddle-node (SN)] is investigated here for the proposed and original models.

Fig. 5 shows the phase plane graphs of the original and modified models for the case of tonic spiking. By changing the value of bifurcation parameter (input current in this case), the number of fixed points changes. For small values of the stimulation current, as shown in Fig. 5(a), there are two fixed points in the original model at $(-55.77,\ 59.30)$ and $(-47.21,\ 93.54)$. The eigenvalues for these points are: $(-0.095,\ -0.030)$ and $(0.418,\ -0.024)$, respectively. The left point, that has two negative eigenvalues, is a stable node point. The next point with one positive and one negative eigenvalue is a saddle point [54]. By further increasing the stimulation current the saddle and node points get closer to each other as shown in Fig. 5(b). The two equilibrium points come together and at the same time an invariant circle is formed [Fig. 5(c)]. In other words, a stable limit cycle appears via a saddle-node on invariant circle (SNIC) bifurcation. Because of presence of the invariant circle, and quite close to the bifurcation point, the neuron generates very low frequency spikes. This scenario also happens in the proposed models as shown in Fig. 5(d), (e), and (f). The Fig. 6 shows the saddle-node bifurcation of the PLE and PWL approximations for the same AdEx model taken from Gomar *et al.* [37]. The shape of the nullclines determine the frequency and shape of the time domain signals. In the PWL and PLE models for the same stimulation current the coordinate of the fixed points are different comparing with original model [Fig. 6 column (a), (b)]. Furthermore, for some stimulation values there are more than two fixed points or there is no distinctive annihilation point [Fig. 6 column (b)]. This also accounts for PLE model [see the zigzag lines in nullclines of Fig. 6 column (c)].

C. Network Behavior

To analyze the network behavior of the proposed and original models, at first it is assumed that 100 synapses are attached to

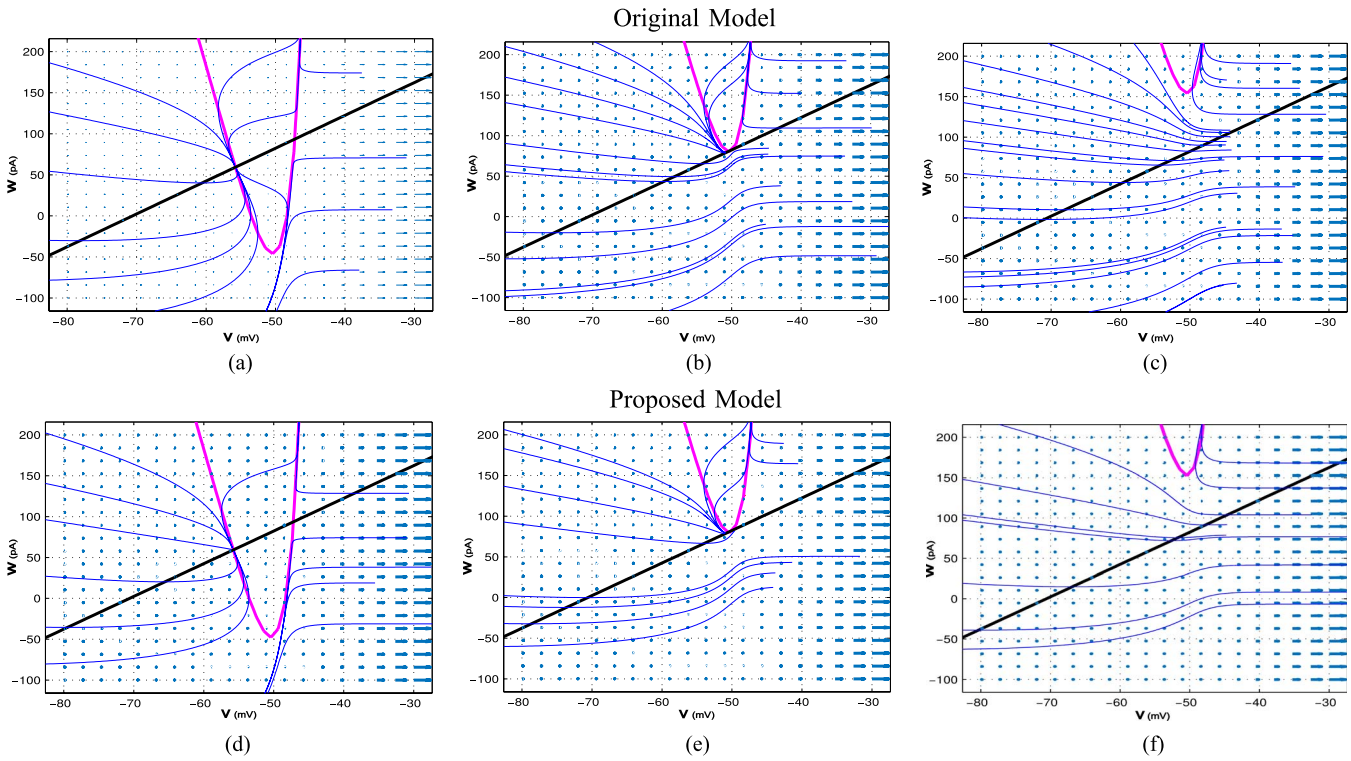


Fig. 5. Saddle-node bifurcation in the proposed and original models. Rows (a)–(c), and (d)–(f) show bifurcation for the original the proposed models, respectively. Similar to the original model, these systems have two interaction points (saddle and node) and therefore similar bifurcations. The first row shows the state of the models for low injected current; the second row shows the state of the models for the reheobase current (saddle node annihilation point). If the injected current is further increased the state of the models will be similar to the third row.

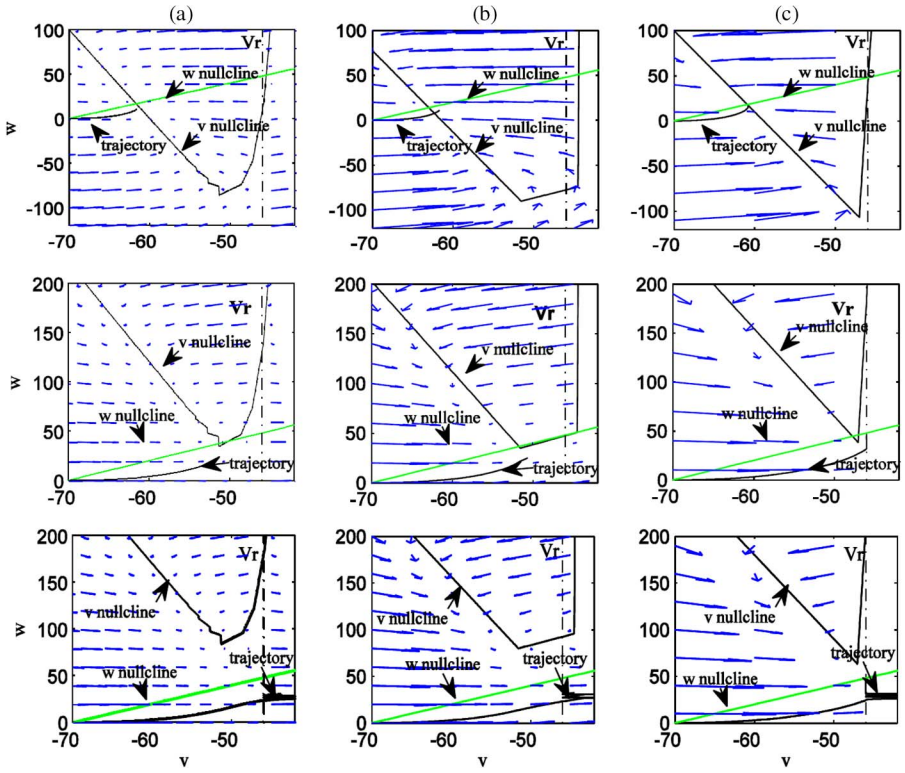
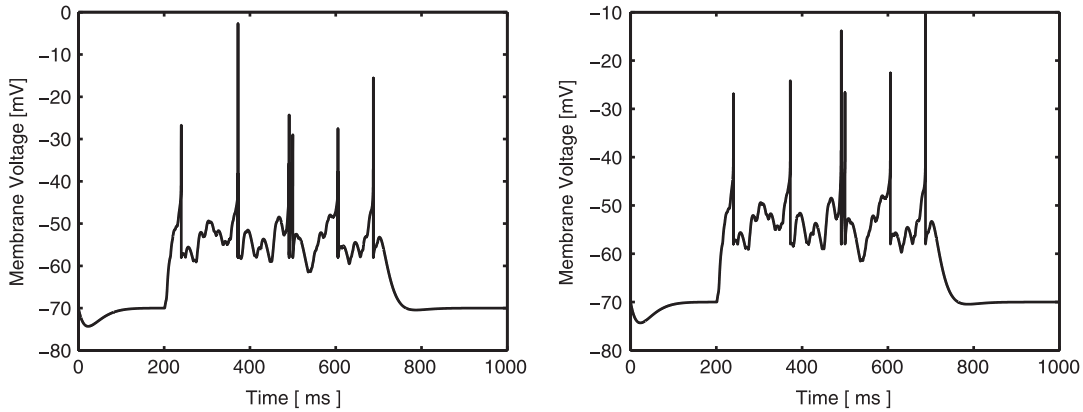


Fig. 6. Saddle-node bifurcation for the modified proposed adaptive exponential integrate and fire neuron models by Gomar *et al.* [37].

the neuron, with each pre-synaptic neuron firing with a Poisson process of rate $f_{\text{rate}} = 2$ Hz between 200 ms and 700 ms. Fig. 7 presents simulation results for both original [Fig. 7(a)] and the CORDIC model [Fig. 7(b)]. As indicated by the figure, the proposed model has a very close response to the original model, and there is no notable difference between responses.



(a) Original Model (b) Proposed Model

Fig. 7. Computer simulation of the neuron models responses to the input of 100 synapses each one with a pre-synaptic neuron firing with a Poisson process of rate f rate $= 2$ Hz between time 200 ms and 700 ms. (a) Original model. (b) CORDIC model.

In many neural network learning algorithms, like the STDP [55], timing of the spikes is one of the most important factors and the shape or amplitude of them is dispensable (although some researchers challenge this, see [56] for further arguments). Therefore, in many applications it is enough to just keep the spike timings. In this regard, to check the proposed model accuracy, a network of 1000 randomly connected neurons (with identical tonic spiking models) are simulated. Inspired by the mammalian cortex, the ratio of the excitatory to inhibitory neurons is selected to be 4 to 1. The input current of a neuron is the sum of a random current and the current generated by other spiking neurons multiplied by the associate synapse weight as

$$I_j(t) = I_r + S_{ji} \times \text{fired}_i \tag{19}$$

where the fired_i is a Boolean variable, which is equal to 1 if neuron i fires and 0 otherwise. The results are presented as raster diagrams in Fig. 8. In these diagrams, each dot stands for a specific neuron spiking at a specific time [57]. The network information is contained in the spikes 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 concentration. Even though neurons are coupled randomly and there is no synaptic plasticity, inspecting spikes' raster diagrams show the bursting synchronization for the active neurons. The neurons are self-organized into groups and show a generic rhythmic behavior [13]. There are unsynchronized neurons in these diagrams, yet synchronization is the dominant behavior. Fig. 8(a) and (b) show the raster diagrams for the original model and the proposed one. They have differences in a few details, but in general, there is no significant differences between the two figures.

It is widely believed that the rate in which a neuron fires spikes, is the core mechanism of processing information in biological neural networks. Accordingly, for comparing two models, the spike count rate for a random neuron in the previous spike raster is depicted in Fig. 9. It can be calculated simply by counting the number of spikes in the time duration (Δt) and dividing it by the length of the time duration (Δt), which results in a temporal mean computing. As shown, the figures are quite similar for a random neuron.

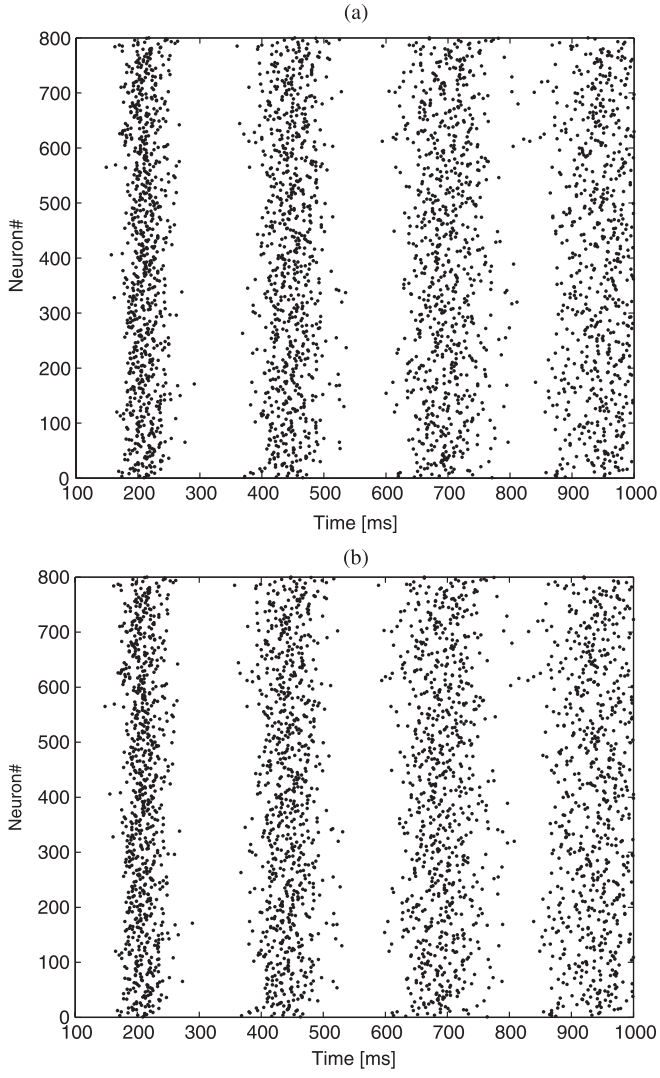


Fig. 8. Raster diagram for computer simulation of the randomly coupled network of the original and proposed model. Each dot stands for a specific neuron, spiking at a specific time. (a) Original model; (b) proposed model.

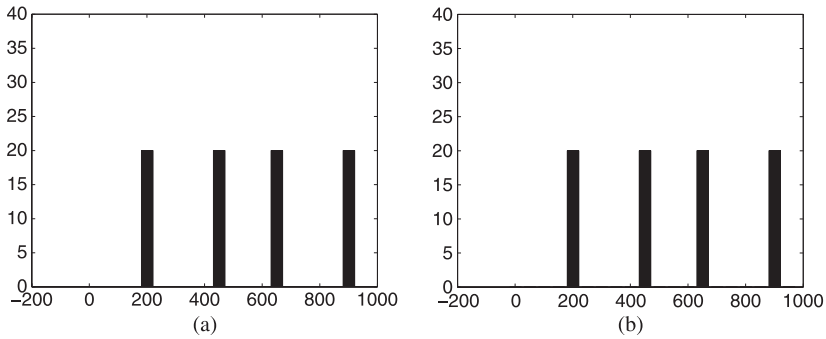


Fig. 9. Spike count rate of a random neuron from the raster Fig. 8. Time interval $\Delta t = 50$ ms. (a) Original model; (b) proposed model.

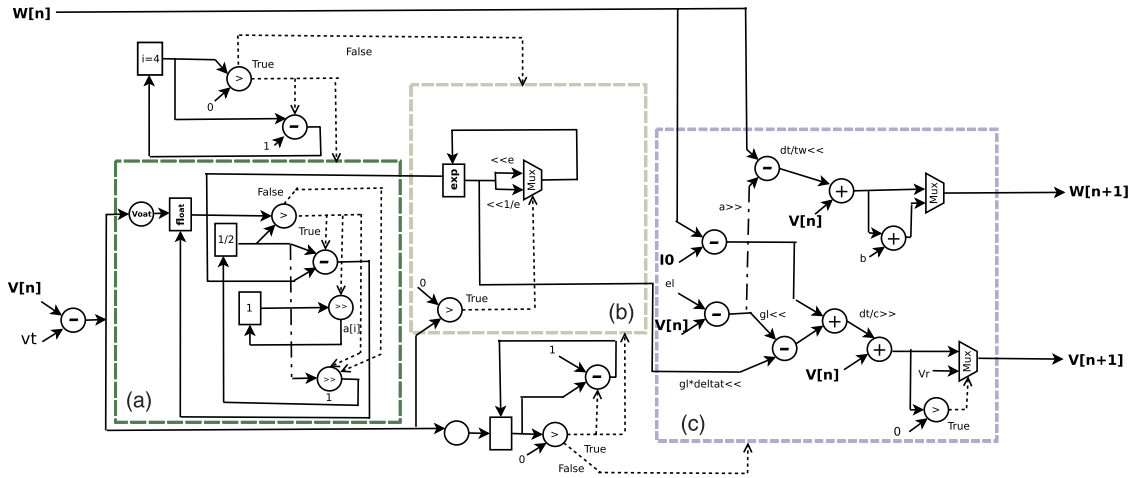


Fig. 10. Control data flow graph for digital implementation of the CORDIC model. Block (a) calculates the fraction part of the exponential function and block (b) is responsible for the integer part. With the exponential function calculated by the previous blocks, block (c) calculates (1) and checks the reset condition in (2). The plain line shows the flow of data and the dashed lines indicate the jumps and decision signals.

IV. Neuron On Hardware

A. Architecture

This section presents a hardware implementation of the proposed CORDIC AdEx neuron and its realization on FPGA as a proof of concept. Using fixed-point arithmetic, the model is implemented as a digital module. For digital computation, (1) and (3), are discretized using the Euler method as

$$\begin{aligned} v[n+1] &= v[n] + dt(1/c(-gl(v[n]-el)+\cdots \\ &gl\Delta t(\exp\_\text{cordic}((v[n]-vt)/\Delta t)) + I[n] - w[n])) \\ w[n+1] &= w[n] + dt(1/tw(a(v[n]-el) - w[n])) \end{aligned} \quad (20)$$

where exp_cordic is the CORDIC exponential function computer. The Euler method is an explicit and simple method for numerical solution of the ODEs with given initial conditions, which match our requirements, that is reducing the implementation costs. Although the method has a bad reputation for its stability, with the help of small step sizes and a reset equation, the method produces steady output waveforms both in the simulation and hardware realization.

To avoid multipliers, constant number multiplications are realized on hardware using "**shift**" and "**add**" operations. Therefore, constant coefficients in the modified model were approximated with the series of sum of 2^n numbers, where $n = 1, 2, 3, \ldots, k$. Each number shifts the multiplicand. The multiplication product is the sum of these shifted multiplicands. Selecting the value of k is a trade-off between simplicity and precision. This method reduces the model accuracy but it is very efficient in comparison with multipliers. The control data flow graph (CDFG) [58] of the model is shown in Fig. 10.

In this CDFG, each circle represents a basic operation block, and each rectangle is a register. There are three major blocks in the figure. The **A** block is responsible for calculating the exponential fraction, and **B** calculates the exponential integer part, where the last block, **C**, is responsible for solving the equation using Euler method.

In the CDFG diagram "«" represents arithmetic shift operation. For instance "gl«" means shift the variable by gl, which gl is written in the form of sum of 2^n numbers like $(1/2) + (1/8) + (1/16)$. To avoid extra adders in each step, shift operations were spread over steps wherever possible. Since it does not require any multiplier and calculates functions using only **add** and **shift** operations, it is expected that the operation frequency would not be affected considerably.

At the next step, the optimum bit width of the arithmetic units for the minimum hardware cost have been calculated. To determine the minimum required bit width, the range of variables, the maximum number of shift operations and minimum precision for the coefficients should be taken into account. For the fraction part of the variables, 8 bits are dedicated as the minimum acceptable precision. For calculation of "v," ultimate number of 15 bit arithmetic right shift operation is required. So,

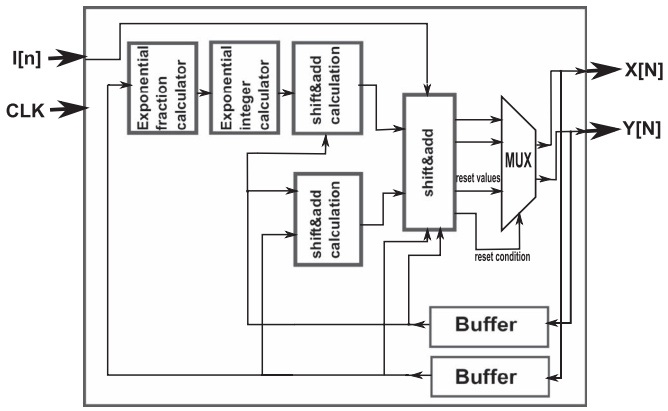


Fig. 11. General overview of the implementation architecture. (To avoid any confusion, detailed signal wiring and the clock signal tree are not presented.)

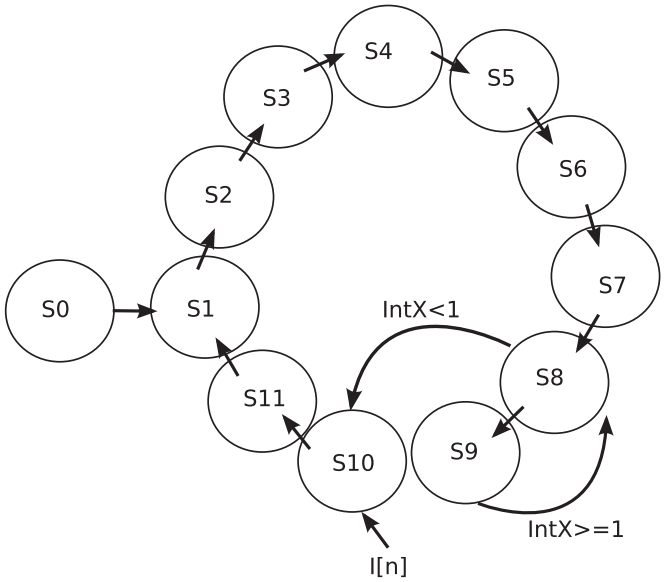


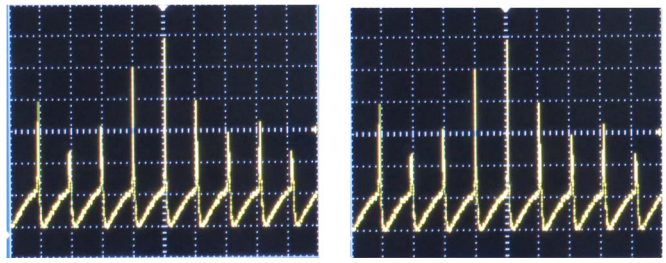
Fig. 12. State machine diagram of the neuron circuit.

overall 23 bits word length is selected for the fraction part of the variables. The worst number of arithmetic left shift operations is 7 times, which also shifts the variable "v" that is considered to have 7 bits integer part. Thus, 14 bits are considered for the integer parts of the variables. In total, to avoid precision loss and overflow, the maximum number of 37 bits has been used, which dedicates 23 bits for the fraction and 14 bits for the integer parts. The time step (dt) is chosen as 1/128 because it can simply be implemented by 7 arithmetic right shifts.

A pipelined architecture for implementation of the proposed model is presented in Fig. 11. Two buffers (V buffer and W buffer) are utilized to store the output of v and w modules. These buffers are used both in the first step as well as in the whole scheduling sequence until new values of v and w are produced.

B. HDL Description and FPGA Implementation

The CDFG in the Fig. 10 is described as HDL code (Verilog) using a finite state machine (FSM) with 12 states as shown in Fig. 12. States 2 up to 9 are responsible for calculating the exponential function [Fig. 10 block (a), (b)] and states 10–12 are for solving the Euler method in (20) [Fig. 10 block (c)].



(a) (b)

Fig. 13. The 5 bits of the implemented models converted to analog using VGA on board DAC and output signals observed on oscilloscope. (a) Tonic spiking. (b) Regular bursting. (time scale $= 250$ μs, volt. scale $= 250$ mv.

TABLE II
DEVICE UTILIZATION OF THE XILINX SPARTAN-6 XC6LX9 BASED ON SPEED OPTIMIZATION GOAL

Resources	Slice Registers	Slice LUT's	Max Speed (MHz)
Tonic spiking	829	1221	134.3
Regular bursting	864	1246	128.7
Total Available	11,440	5,572	250

The first state assigns the initial values of V and W. State 2, calculates the value of $(v[n] - vt)/\Delta t)$ and stores it in a register (X). During the same state, the integer part of X will be stored in the register Int X and the register containing the final value of $\exp((v[n] - vt)/\Delta t)$ (Exp X) will be reset to one. Likewise, in state 3 the floating part of X is stored in a register (Float X) and the register that will be used to compare the floating part with the power of two factors (PowerOfTwo) will be set to 1/2.

In the next state the PowOfTwo will be compared with the Float X and if Float $X >$ PowOfTwo,Exp X will be shifted according to the value of $a[1]$ as explained in the previous subsection. Also the PowOfTwo will be arithmetic shifted to the right. During the next three steps, the system performs similar operations (considering $n = 4$ as discussed in Section II).

At state 8, the circuit compares the value Int x with 1 and based on that decides the next state to be either state 9 or 10. State 9 shifts the Exp X by e (Euler's number) or $(1/e)$, depending on whether X is positive or negative, respectively. Then the value of Int X is decreased by one and goes to state 8 again to make a decision on the next state. State 10 calculates $dv[n]$, and $dw[n]$ with **add** and **shift**, where the result is Exp X, $V[n]$, $w[n]$, and stimulation current $I[n]$. State 11 performs 7 arithmetic shifts on $dv[n]$ and $dw[n]$ with respect to dt, which has been chosen as 1/128 and calculates $v[n+1]$ and $w[n+1]$. In state 11, the reset condition will also be checked, the output register is updated with new values of v and w, and the next state is set as state 1. As determined in the previous section, all registers have 37 bits width except the output registers, which have 15 bits width.

The Verilog description of the system has been simulated using Modelsim. To prove validation of the model, the HDL code is also synthesized by XILINX ISE, and has been deployed on a XILINX Spartan-6 FPGA with XC6SLX9 FPGA development board, which utilizes 45 nm process technology.

Fig. 13 shows oscilloscope output photographs of the physically implemented proposed model on the FPGA for two

TABLE III
COMPARISON BETWEEN PROPOSED APPROACH AND PREVIOUSLY PUBLISHED WORKS

Refrence	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
Hayati et al.	476	856	135	0	3.7	-	Virtex-II Pro XC2VP30
Grassia et al.	646	1048	105	22	-	-	Virtex-5 XC5VLX50
This work	829	1221	134.3	0	0.04	0.39	Spartan-6 XC6SLX9

test cases: tonic spiking and regular bursting. The stimulation current is applied as in Fig. 4, and 5 bits [10:6] out of the 15-bit output registers are selected and connected to the on board digital to analog converter (DAC).

C. Results and Discussion

The device utilization for the physical implementation of the model is summarized in Table II. These results prove the efficiency of the proposed neuron hardware. Furthermore, Table III presents the FPGA device utilization, the error NRMSD, and Errt of the proposed hardware implementation in comparison with previously published works in which spiking neural networks are implemented on similar hardware platforms. Please note that in some cases the FPGA devices and synthesizer versions that have been used for implementation are different, therefore, the results presented in this table must be considered relatively. Also, as shown in the Table II by taking advantage of multiplexing operands and sharing resources, the utilized FPGA resources can be reduced.

V. CONCLUSION

The exponential term in adaptive exponential integrate and fire neuron (AdEx) have been calculated using the CORDIC algorithm. The simulation and implementation results prove that the proposed model follows the same dynamical behavior in both the time and phase domains as AdEx neuron. To draw a quantitative comparison, different errors were calculated. Both models were also similar in the dynamical and bifurcation point of view. The calculated errors and dynamical analysis also confirm the resemblance of the models. With the benefit of the CORDIC method, which only uses the shift&add operation, digital hardware was designed to implement the CORDIC AdEx neuron. The model was implemented on FPGA and the output waveforms were observed on an oscilloscope. FPGA device utilization and speed also verify the efficiency of the designed hardware. This model can be used for different types of neurons by changing parameter values, producing several types of neurons behaviors: tonic spiking, regular bursting, etc.

REFERENCES

[1] J. Misra and I. Saha, "Artificial neural networks in hardware: A survey of two decades of progress," *Neurocomputing*, vol. 74, no. 1–3, pp. 239–255, 2010.

[2] W. Maass, G. Schnitger, and E. Sontag, "On the computational power of sigmoid versus boolean threshold circuits," in *Proc. 32nd Annu. Symp. Found. Comput. Sci.*, Oct. 1991, pp. 767–776.

[3] W. Gerstner, R. Kempter, J. L. van Hemmen, and H. Wagner, "A neuronal learning rule for sub-millisecond temporal coding," *Nature*, vol. 383, no. 6595, pp. 76–78, 1996.

[4] R. Eckhorn, R. Bauer, W. Jordan, M. Brosch, W. Kruse, M. Munk, and H. Reitboeck, "Coherent oscillations: A mechanism of feature linking in the visual cortex?" *Biol. Cybern.*, vol. 60, no. 2, pp. 121–130, 1988.

[5] C. von der Malsburg and W. Schneider, "A neural cocktail-party processor," *Biol. Cybern.*, vol. 54, no. 1, pp. 29–40, 1986.

[6] F. Ponulak and A. Kasinski, "Introduction to spiking neural networks: Information processing, learning and applications," *Acta Neurobiologiae Experimentalis*, vol. 71, no. 4, pp. 409–433, 2011.

[7] A. Grüning and S. M. Bohte, "Spiking neural networks: Principles and challenges," in *Proc. 22nd Eur. Symp. Artif. Neural netw., comput. Intell. Mach. Learn. (ESANN)*, Oct. 2014, pp. 1–10.

[8] J. Vreeken, "Spiking neural networks: An introduction," Artificial Intelligence Laboratory, Intelligent Systems Group, Univ. Utrecht, 2003.

[9] R. FitzHugh, "Impulses and physiological states in theoretical models of nerve membrane," *Biophys. J.*, vol. 1, pp. 445–466, Jul. 1961.

[10] C. Morris and H. Lecar, "Voltage oscillations in the barnacle giant muscle fiber," *Biophys. J.*, vol. 35, no. 1, pp. 193–213, 1981.

[11] H. R. Wilson, "Simplified dynamics of human and mammalian neocortical neurons," *J. Theor. Biol.*, vol. 200, no. 4, pp. 375–388, 1999.

[12] R. M. Rose and J. L. Hindmarsh, "The assembly of ionic currents in a thalamic neuron i. the three-dimensional model," *Proc. Roy. Soc. B, Biol. Sci.*, vol. 237, no. 1288, pp. 267–288, 1989.

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

[14] R. Brette, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity," *J. Neurophysiol.*, vol. 94, no. 5, pp. 3637–3642, 2005.

[15] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *Bull. Math. Biol.*, vol. 52, no. 1–2, pp. 25–71, 1990.

[16] "Adaptive exponential integrate and fire (adex) neuron model," 2016. [Online]. Available: http://www.digicortex.net/node/33

[17] L. Badel, S. Lefort, R. Brette, C. C. H. Petersen, W. Gerstner, and M. J. E. Richardson, "Dynamic i-v curves are reliable predictors of naturalistic pyramidal-neuron voltage traces," *J. Neurophysiol.*, vol. 99, no. 2, pp. 656–666, 2008.

[18] W. Gerstner and R. Brette, "Adaptive exponential integrate-and-fire model," *Scholarpedia*, vol. 4, no. 6, p. 8427, 2009.

[19] S. Davies, "Learning in the spiking neural networks," Ph.D. dissertation, Univ. Manchester, Manchester, U.K., 2012.

[20] "The human brain project," 2016. [Online]. Available: https://www.humanbrainproject.eu

[21] "Bluebrain | epfl," 2016, last visited 2012-03-18. [Online]. Available: http://bluebrain.epfl.ch

[22] "Cognitive computation project," 2016. [Online]. Available: http://ibm.com

[23] "Nest simulator | the neural simulation tool," 2016. [Online]. Available: http://www.nest-simulator.org/

[24] "The brain spiking neural network simulator," 2016. [Online]. Available: http://briansimulator.org/

[25] "Nemo," 2016. [Online]. Available: http://nemosim.sourceforge.net/

[26] A. Joubert, B. Belhadj, O. Temam, and R. Heliot, "Hardware spiking neurons design: Analog or digital?" in *Proc. IJCNN*, Jun. 2012, pp. 1–5.

[27] A. S. Cassidy, J. Georgiou, and A. G. Andreou, "Design of silicon brains in the nano-cmos era: Spiking neurons, learning synapses and neural architecture optimization," *Neural Netw.*, vol. 45, pp. 4–26, 2013.

[28] O. Sharifipoor and A. Ahmadi, "An analog implementation of biologically plausible neurons using CCII building blocks," *Neural Netw.*, vol. 36, pp. 129–135, 2012.

[29] S. Millner, A. Grübl, K. Meier, J. Schemmel, and M. Olivier Schwartz, "A VLSI implementation of the adaptive exponential integrate-and-fire neuron model," in *Advances in Neural Information Processing Systems 23*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds. Red Hook, NY, USA: Curran Assoc., 2010, pp. 1642–1650.

[30] S. Brink, S. Nease, P. Hasler, S. Ramakrishnan, R. Wunderlich, A. Basu, and B. Degnan, "A learning-enabled neuron array ic based upon transistor channel models of biological phenomena," *IEEE Trans. Biomed. Circuits Syst.*, vol. 7, no. 1, pp. 71–81, Feb. 2013.

[31] Y. Yamashita and H. Torikai, "A novel pwc spiking neuron model: Neuron-like bifurcation scenarios and responses," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 11, pp. 2678–2691, Nov. 2012.

[32] J. A. Starzyk and Basawaraj, "Memristor crossbar architecture for synchronous neural networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 8, pp. 2390–2401, Aug. 2014.

[33] Y. Wang and S. C. Liu, "A two-dimensional configurable active silicon dendritic neuron array," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 9, pp. 2159–2171, Sep. 2011.

[34] G. Indiveri, B. Linares-Barranco, T. J. Hamilton, A. van Schaik, R. Etienne-Cummings, T. Delbruck, S.-C. Liu, P. Dudek, P. Häfliger, S. Renaud, J. Schemmel, G. Cauwenberghs, J. Arthur, K. Hynna, F. Folowosele, S. SAÏGHI, T. Serrano-Gotarredona, J. Wijekoon, Y. Wang, and K. Boahen, "Neuromorphic silicon neuron circuits," *Front. Neurosci.*, vol. 5, no. 73, pp. 1–23, 2011.

[35] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proc. IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[36] H. Soleimani, A. Ahmadi, and M. Bavandpour, "Biologically inspired spiking neurons: Piecewise linear models and digital implementation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 12, pp. 2991–3004, Dec. 2012.

[37] S. Gomar and A. Ahmadi, "Digital multiplierless implementation of biological adaptive-exponential neuron model," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 4, pp. 1206–1219, Apr. 2014.

[38] T. Matsubara, H. Torikai, and T. Hishiki, "A generalized rotate-and-fire digital spiking neuron model and its on-fpga learning," *IEEE Trans. Circuits Syst. II, Express Briefs*, vol. 58, no. 10, pp. 677–681, Oct. 2011.

[39] F. Grassia, T. Levi, T. Kohno, and S. Saïghi, "Silicon neuron: digital hardware implementation of the quartic model," *Artif. Life Robot.*, vol. 19, no. 3, pp. 215–219, 2014.

[40] D. Neil and S. C. Liu, "Minitaur, an event-driven fpga-based spiking network accelerator," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 12, pp. 2621–2628, Dec. 2014.

[41] A. Linares-Barranco, G. Jimenez-Moreno, B. Linares-Barranco, and A. Civit-Balcells, "On algorithmic rate-coded aer generation," *IEEE Trans. Neural Netw.*, vol. 17, no. 3, pp. 771–788, May 2006.

[42] M. Hayati, M. Nouri, S. Haghiri, and D. Abbott, "Digital multiplierless realization of two coupled biological morris-lecar neuron model," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 7, pp. 1805–1814, Jul. 2015.

[43] N. Yildiz, E. Cesur, K. Kayaer, V. Tavsanoglu, and M. Alpay, "Architecture of a fully pipelined real-time cellular neural network emulator," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 1, pp. 130–138, Jan. 2015.

[44] A. Cassidy and A. G. Andreou, "Dynamical digital silicon neurons," in *Proc. IEEE Biomed. Circuits Syst. Conf.*, Nov. 2008, pp. 289–292.

[45] M. Berzish and B. Tripp, "A digital hardware design for real-time simulation of large neural-system models in physical settings," *BMC Neurosci.*, vol. 15, no. 1, 2014.

[46] N. Fourcaud-Trocmé, D. Hansel, C. van Vreeswijk, and N. Brunel, "How spike generation mechanisms determine the neuronal response to fluctuating inputs," *J. Neurosci.*, vol. 23, no. 37, pp. 11628–11640, 2003.

[47] R. Naud, N. Marcille, C. Clopath, and W. Gerstner, "Firing patterns in the adaptive exponential integrate-and-fire model," *Biol. Cybern.*, vol. 99, no. 4/5, pp. 335–347, 2008.

[48] J. E. Volder, "The cordic trigonometric computing technique," *IRE Trans. Electron. Comput.*, vol. EC-8, no. 3, pp. 330–334, Sep. 1959.

[49] J. Walther, "The story of unified cordic," *J. VLSI Signal Process. Syst. Signal, Image, Video Technol.*, vol. 25, no. 2, pp. 107–112, 2000.

[50] J.-M. Muller, *Elementary functions*. Boston, MA, USA: Birkhauser, 2006.

[51] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecast.*, vol. 22, no. 4, pp. 679–688, 2006.

[52] E. M. Izhikevich, *Dynamical Systems in Neuroscience*. Cambridge, MA, USA: MIT Press, 2007.

[53] L. Hertäg, J. Hass, T. Golovko, and D. Durstewitz, "An approximation to the adaptive exponential integrate-and-fire neuron model allows fast and predictive fitting to physiological data," *Front. Comput. Neurosci.*, vol. 6, pp. 1–22, 2012.

[54] J. Touboul and R. Brette, "Dynamics and bifurcations of the adaptive exponential integrate-and-fire model," *Biol. Cybern.*, vol. 99, no. 4/5, pp. 319–334, 2008.

[55] A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biol. Cybern.*, vol. 98, no. 6, pp. 459–478, 2008.

[56] B. M. Kampa, J. J. Letzkus, and G. J. Stuart, "Dendritic mechanisms controlling spike-timing-dependent synaptic plasticity," *Trends Neurosci.*, vol. 30, no. 9, pp. 456–463, 2007.

[57] S. M. Kosslyn and R. A. Andersen, *Frontiers in Cognitive Neuroscience*. Cambridge, MA, USA: MIT Press, 1992.

[58] S. P. Mohanty, *Low-Power High-Level Synthesis for Nanoscale CMOS Circuits*. New York, NY, USA: Springer, 2008.

Moslem Heidarpour received the B.Sc. degree in electrical engineering and M.Sc. degree in electronic engineering from the Department of Electrical Engineering, Razi University, Kermanshah, Iran, in 2012 and 2014. His research interests include analog and digital electronic circuit design and optimization, bio-inspired computing, neuromorphic and integrated circuit design.

Arash Ahmadi (M'04–SM'16) received the B.Sc. and M.Sc. degrees in electronics engineering from Sharif University of Technology and Tarbiat Modares University, Tehran, Iran, in 1993 and 1997, respectively, and the Ph.D. degree in electronics from the University of Southampton, U.K., in 2008. He was with Razi University, Kermanshah, Iran, as a Faculty Member. From 2008 to 2010, he was a Fellow Researcher with the University of Southampton. He is currently an Associate Professor in the Electrical Engineering Department, Razi University, and Visiting Scholar at University of Windsor, Canada. His current research interest includes neuromorphic, hardware implementation of signal processing systems, bio-inspired computing, memristors, hardware security, and IoT.

Rashid Rashidzadeh (M'04–SM'13) received the B.S.E.E. degree from Sharif University of Technology, Tehran, Iran, and the M.A.Sc. and Ph.D. degrees in electrical engineering from University of Windsor, Windsor, ON, Canada, in 2003 and 2007, respectively. He has a track record of successful collaboration with industry. Dr. Rashidzadeh has supervised many industry projects, the systems designed by his research team have entered the market successfully. He is currently a Manager of Research Centre for Integrated Microsystems (RCIM) and an Adjunct Professor with the Electrical and Computer Engineering Department at the University of Windsor. His current research interests include Design-for-Test (DfT) methodologies for integrated circuits, radio frequency identification and smart sensors for Internet of Things (IoT). Dr. Rashidzadeh was the recipient of the Excellence in Research, Scholarship and Creative Activity Award at the University of Windsor in 2015. He is the Chair of the IEEE Circuits and Systems and Computer Societies Chapter in Region Seven.