Low-Power Resource-Efficient FPGA Implementation of Modified FitzHugh–Nagumo Neuron for Spiking Neural Networks

Reza Badiei, Somayeh Timarchi, and Alireza Zakaleh

***Abstract*—The primary goals of neuromorphic engineering are to study, simulate, model, and implement neural behavior of the human brain. In this work, we propose a modified version of the original FitzHugh-Nagumo (FHN) neuron model in which the nonlinear term is replaced with a power-of-two-based approximation. The modification eliminates the need for multipliers, reducing hardware resource utilization while maintaining high fidelity in reproducing the dynamic behaviors of the original model. To validate the proposed model, we conduct dynamic behavior analysis, error evaluation, and network behavior simulation, demonstrating that it accurately reproduces the key characteristics of the FHN model with minimal error. An efficient digital hardware solution for implementing neurons optimized for large-scale Spiking Neural Networks (SNNs), leveraging resource-sharing techniques and pipelining strategies, is presented. The design is described using the VHSIC Hardware Description Language (VHDL), simulated and synthesized in Vivado, and implemented on a Xilinx Zynq Field-Programmable Gate Array (FPGA). Experimental results demonstrate that the proposed model achieves a normalized RMSE of 0.36, while utilizing only 0.38% of the available resources, including 0.27% of slice LUTs and 0.16% of registers. Additionally, it operates at a frequency of 255 MHz while consuming only 29 mW of power. Moreover, the FPGA implementation of our proposed model requires fewer resources and lower power consumption compared to previous works, while maintaining a comparable error rate.**

***Index Terms*—Spiking Neural Networks, FitzHugh-Nagumo neuron, Neuromorphic computing, FPGA.**

I. Introduction

ONE of the central focuses of neuroscience research is exploring neuron models and their interactions within the human brain. Various spiking neuron models have been proposed to mimic the behavior of biological neurons [1]. Many of these models are simplified deviations of the Hodgkin-Huxley (HH) model [2], which, despite its high accuracy in reproducing diverse spiking patterns, remains computationally intensive. For instance, the FitzHugh-Nagumo (FHN) model [3], [4], a simplified version of the HH model, from which several hybrid neuron models have been derived, is a two-coupled differential equation model with only one polynomial term. Compared to the HH model, the FHN model offers a better trade-off between biological realism and computational simplicity, making it suitable for Spiking Neural Network applications [5], [6]. Unlike the Leaky Integrate-and-Fire model, FHN reproduces complex behaviors such as bursting [7] and synchronization, crucial for modeling realistic neural dynamics and higher-level brain functions [8].

Numerous studies have focused on hardware implementations of the FHN model [5], [6], [7], [8], [9], [10], while some have addressed its behaviorally complex variants, such as the FitzHugh-Nagumo-Morris-Lecar model [11] and the FitzHugh-Rinzel model [12]. In [5], resource efficiency was improved by replacing the nonlinear term with two alternatives: a piecewise linear (PWL) approximation and a formulation based on the hyperbolic sine (sinh) function. Behavioral simulations of postsynaptic neural activity regulation and synaptic transmission mechanisms were conducted in [6]. A multiplier-less burst-mode FHN model, implemented in [7], used linear estimation to reduce hardware cost. Studies in [8] and [9] employed the coordinate rotation digital computer (CORDIC) algorithm to achieve multiplication-free designs while maintaining high accuracy. An approximation algorithm was introduced in [10] to improve precision and reduce cost by converting computational data into floating-point numbers. Overall, these studies on spiking neurons use shift-and-add techniques as a low-cost alternative to multiplication [5], [6], [7], [8], [9], [10], [11], [12], [13]. However, despite these efforts, they still exhibit high resource and power requirements, with limited optimizations for reducing hardware costs while maintaining performance. Therefore, by using approximate computing techniques [14], [15], this work proposes replacing the nonlinear term with a power-of-two-based function to achieve a hardware-friendly, low-power implementation.

The brief is organized as follows. Section II presents the original FHN model, while Section III introduces the proposed model. An assessment of the dynamic behavior, error analysis, and network behavior of the proposed model is provided in Section IV. Section V covers the hardware design and implementation, followed by the results of the implementation. Finally, the brief is concluded in Section VI.

II. Background

The Fitzhugh-Nagumo neuron model was initially proposed by Richard Fitzhugh in 1961 [3], and in 1962, Nagumo

Received 8 June 2025; revised 18 August 2025; accepted 23 September 2025. Date of publication 1 October 2025; date of current version 4 November 2025. This brief was recommended by Associate Editor C. Meinhardt. *(Corresponding author: Somayeh Timarchi.)*

Reza Badiei and Alireza Zakaleh are with the Faculty of Electrical Engineering, Shahid Beheshti University, Tehran 1983969411, Iran (e-mail: re.badiei@mail.sbu.ac.ir; a.zakaleh@sbu.ac.ir).

Somayeh Timarchi is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, E1 4NS London, U.K. (e-mail: s.timarchi@qmul.ac.uk).

Digital Object Identifier 10.1109/TCSII.2025.3615935

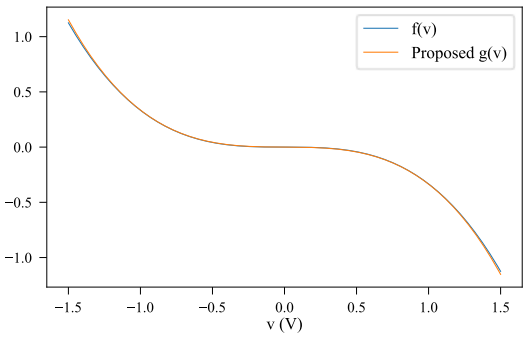

Fig. 1. Matching accuracy between the original and proposed FHN function.

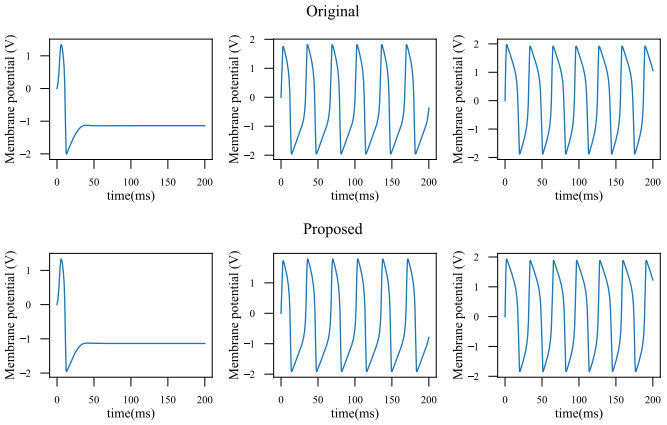

Fig. 2. Spike timing patterns for the original FHN and proposed model.

provided an analog circuit representation of Fitzhugh's model [4]. The dynamic system of the FHN neuron is controlled by the following differential equations [3]:

$$\begin{cases} \dfrac{dv}{dt} = v + f(v) - w + I \\ \tau \dfrac{dw}{dt} = v + a - bw \\ f(v) = -\dfrac{v^3}{3} \end{cases} \tag{1}$$

where v represents the neuron's membrane potential and w serves as the membrane recovery variable. Additionally, a, b, and τ are parameters that influence the spike frequency and shape of the neuron, while I is the applied input current. As shown in the FHN neuron equations, the nonlinear term $f(v)$ may lead to issues in hardware implementation, including increased resource utilization, reduced speed, and overall degradation in performance and efficiency. Therefore, modifying and optimizing the model is the most effective way to address these problems.

III. Proposed Model

In this section, we present a modified version of the FHN model by replacing the nonlinear function $f(v)$ with a simplified alternative $g(v)$, constructed using power-of-two for computational efficiency. Our analysis revealed that the original nonlinear term behaves similarly to hyperbolic functions, which can be effectively approximated using low-cost exponential expressions of the form 2^x. To capture this behavior, we propose an approximation that combines two power-of-2 terms with an additional linear component. This formulation improves the approximation accuracy, especially in the biologically relevant voltage range. This leads to the proposed function:

$$g(v) = r \times v + s \times \left(2^{(-v)} - 2^{(v)}\right) \tag{2}$$

where r and s are coefficients. The linear term $r \times v$ is incorporated to improve accuracy. To ensure that $g(v)$ closely approximates $f(v)$, we determine r and s by finding the values that minimize the root mean square error (RMSE). As a result, the values of the parameters r and s are found to be 4 and 2.89, respectively. By using this approach, the RMSE of our work is only 0.007 within the voltage range of -1.5 V to 1.5 V. Fig. 1 illustrates the high degree of similarity between the proposed function and $f(v)$ across this range, further validating the accuracy of our approximation. Additionally, Fig. 2 demonstrates that our proposed model is similar to the original FHN neuron model in different spike timing patterns.

TABLE I
Comparison of Equilibrium Points Under Different Input Currents

I (mA)	Original (v,w)	Proposed (v,w)
0.1	Spiral sink (-1.137, -0.546)	Spiral sink (-1.135, -0.543)
0.5	Spiral source (-0.804, -0.131)	Spiral source (-0.803, -0.129)
1	Nodal source (0.408, 1.386)	Nodal source (0.404, 1.380)

IV. Theoretical Analysis

This section is dedicated to validating our proposed model through software simulations. The analysis includes dynamic behavior, error evaluation, and network performance to compare the results of the proposed model with those of the original model.

A. Dynamic Analysis

The dynamic analysis of the proposed and original model is presented in this section. The phase portrait is used to describe the qualitative behavior of the dynamic system, such as equilibria and trajectories. The equilibrium points of the original and proposed models, which can be seen in the phase portrait, are computed to validate our model by setting $dv/dt = 0$ and $dw/dt = 0$. The equations of the original model are provided in [9], while the nullclines of the proposed model are written as follows:

$$\begin{cases} \dfrac{dv}{dt} = 0 \Rightarrow v + g(v) + I = w \\ \dfrac{dw}{dt} = 0 \Rightarrow \dfrac{v + a}{b} = w \end{cases} \tag{3}$$

Table I demonstrates that the proposed model closely matches the original FHN model in both the type and values of their equilibrium points. Furthermore, Fig. 3 highlights the similarity in their dynamic behaviors by presenting trajectory diagrams for both models under various applied current levels

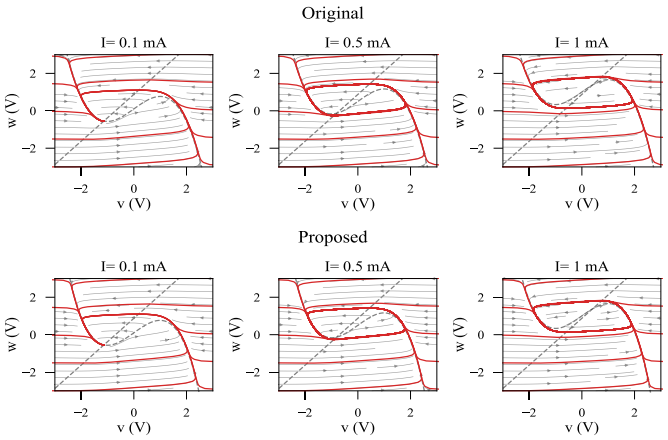

Fig. 3. Phase portraits of the original and proposed models.

TABLE II
ERROR VALUES FOR THE PROPOSED MODEL UNDER VARIOUS INPUTS

I (mA)	MAE	RMSE	NRMSE%
0.1	0.018	0.022	0.665
0.25	0.018	0.022	0.628
0.5	0.029	0.03	0.174
0.75	0.030	0.035	0.181
1	0.031	0.039	0.198
Average	**0.025**	**0.29**	**0.36**

(0.1, 0.5, and 1 mA). As the bifurcation parameter varies, the type and location of the equilibrium points change accordingly, reflecting consistent behavior across both models.

B. Error Analysis

In this section, error analysis is conducted to examine the performance of our proposed model in comparison to the original model. We employ RMSE, Mean Absolute Error (MAE), and Normalized Root Mean Square Error (NRMSE) as evaluation metrics [9]. The RMSE and MAE are used to quantify the deviation between the voltage traces of the two models, while the NRMSE is used to calculate the variance response of the two models. Lower NRMSE values indicate higher similarity in spike patterns between the membrane potential of the original and proposed models.

The RMSE, MAE, and NRMSE values for the proposed model under different inputs with parameters (a = 0.7, b = 0.8, τ = 10) are presented in Table II, which shows that the errors between the proposed model and the FHN model are minimal (with a time step of dt=1ms). These results indicate the accuracy of the proposed model in replicating the behavior of the FHN model, even under diverse input conditions.

C. Network Behavior

To evaluate our proposed neuron in large-scale networks, we simulate a network of 1000 randomly connected neurons with noisy thalamic input currents. The network mimics the mammalian cortex, with 80% excitatory and 20% inhibitory neurons, maintaining a 4:1 excitatory-inhibitory synapse ratio.

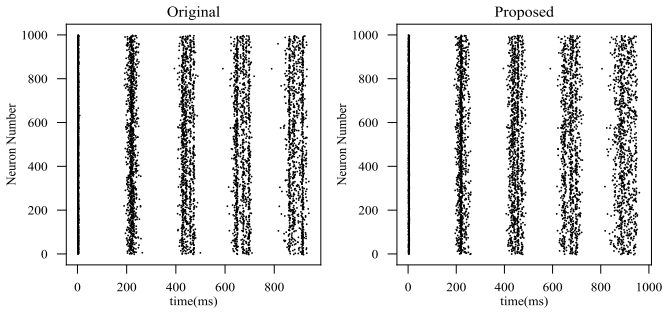

Fig. 4. Raster plot representing the activity of 1000 neurons for the original and proposed models.

Fig. 4 shows raster plots of 1000 neurons for both models, revealing strong similarities in spiking patterns with only minor differences, which confirms the proposed model's ability to replicate the rhythmic behavior of the original model. To measure these minor differences quantitatively, we employ the Mean Relative Error (MRE), which measures the average time difference between corresponding spikes in the proposed and original models presented in [9]. The calculated MRE of 0.28% further validates the proposed model's accuracy in replicating the dynamics of the original model.

V. HARDWARE IMPLEMENTATION

This section presents the FPGA implementation of our proposed neuron model, described using VHDL and verified through the Vivado tool. We detail the digital design process, including Euler-based discretization of the differential equations, the overall architecture of our proposed model, and critical bit-width selection to prevent overflow.

A. Neuron Architecture and System Bit Width

The digital hardware implementation of our neuron model utilizes fixed-point arithmetic, with the differential equations discretized via the Euler method as follows:

$$V[n+1] = V[n] + dt(5V[n] + 2.89(2^{(-V[n])} - 2^{(V[n])}) - W[n] + I[n]) \quad (4)$$

$$W[n+1] = W[n] + dt(V[n] + a - bW[n])/\tau \quad (5)$$

To enhance operating frequency, we adopt a pipelined architecture in our digital implementation, as illustrated in Fig. 5. The computation pipelines for both the membrane potential V and the recovery variable W are constructed using a combination of subtraction, addition, and logical shift operations.

The nonlinear term 2^x in (4) is implemented using the proposed *Pow*2 function illustrated in the following subsection, which leverages shift-based arithmetic instead of multipliers. In line with this design philosophy, all coefficients throughout the architecture are realized using adder–shift combinations to minimize hardware complexity. For instance:

- Multiplication by 5 is implemented as $(x \ll 2) + x$, using two left shifts and one addition.
- The coefficient 2.89 is approximated by 3, realized with $(x \ll 1) + x$.

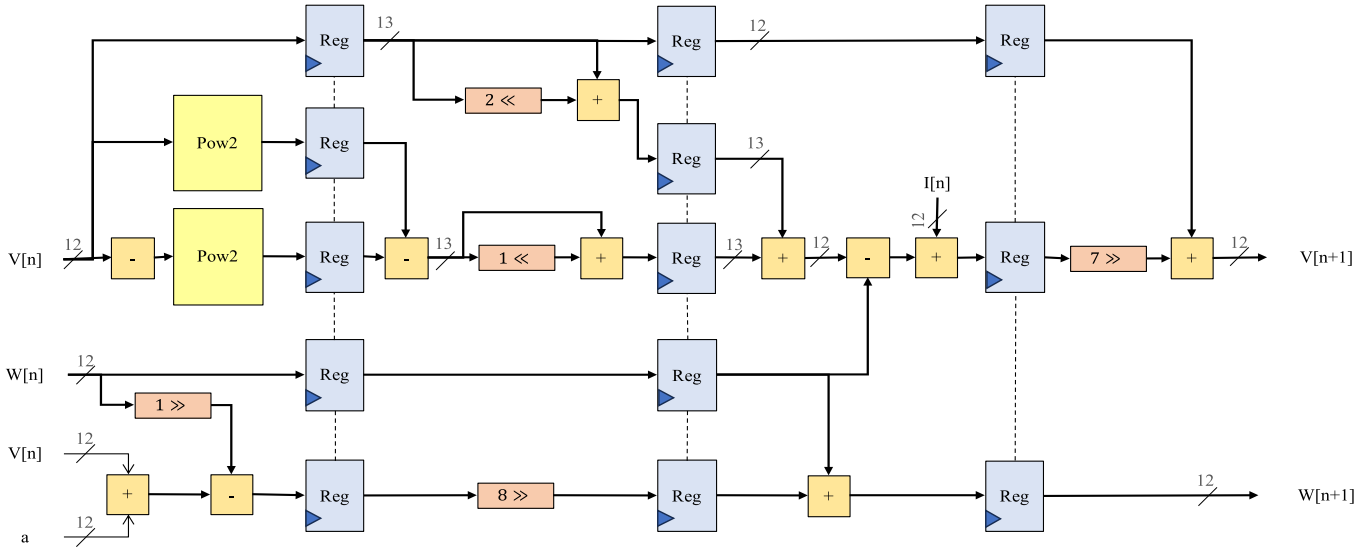

Fig. 5. Proposed multiplierless model using a 4-stage pipeline composed exclusively logical shifters and adders. The symbols '≫' and '≪' denote right and left shift operations, respectively.

• The parameter b in (5), set to 0.5, is implemented via a single right shift ($x \gg 1$).

The time step dt is set to 2^{-7}, efficiently implemented through a 7-bit right shift. Additionally, the time constant τ is chosen as 2, allowing division to be performed with a single right shift. This configuration enables the expression dt/τ in (5) to be computed as a unified 8-bit right shift, further simplifying the control logic.

To ensure both numerical precision and efficient resource usage, we carefully selected the bit-width of variables based on the dynamic range and shift requirements. Accordingly, for the voltage variable V, we allocate: 1 bit for the sign, 2 bits for the integer part, and 9 bits for the fractional part. This results in a total fixed-point width of 12 bits, which provides sufficient accuracy while supporting up to 8-bit right-shift operations without data loss. In rare cases where left shifts may exceed the available range, an additional overflow-protection bit is incorporated, resulting in a total of 13 bits as shown in the figure, ensuring stability across all computational stages.

B. Power-of-Two Unit

To implement the 2^x function, we propose a custom function denoted as $Pow2\,(x)$. The input x, representing the membrane potential V, is constrained within a predefined range. Using fixed-point arithmetic, the input is separated into its integer and fractional components. This decomposition allows us to compute 2^x efficiently: the integer part is handled using simple bit-shift operations, while the power-of-two corresponding to the fractional part is calculated through an optimized approximation. The $Pow2\,(x)$ function is defined as:

$$Pow2\,(x) = (1.\{x\}) \times 2^{\lfloor x \rfloor} \tag{6}$$

where $\lfloor x \rfloor \in \{-2, -1, 0, 1\}$ corresponds to the floor of x and $\{x\}$ shows the fractional part of x. The $Pow2$ function first calculates $1.\{x\}$, then bit-shifts this value by $\lfloor x \rfloor$ to produce the final approximation (e.g., multiplying by 2^1 for $x \in [1, 2)$).

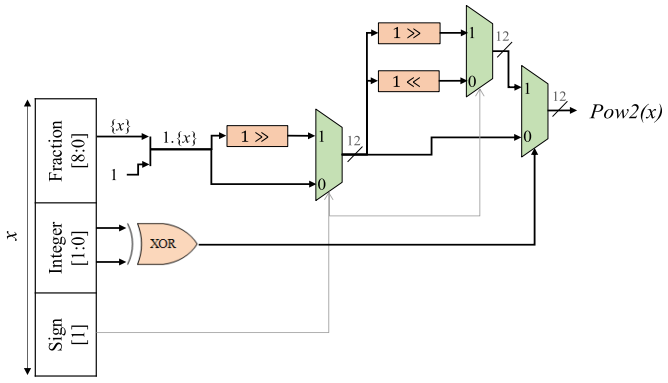

Fig. 6. Hardware implementation for the proposed $Pow2x$ function based on (6). x represents the fractional part of x.

This enables efficient implementation using logical shifters. Based on (6), the hardware implementation of the proposed $Pow2$ function is shown in Fig. 6. First, the circuit computes $1.\{x\}$ by concatenating a '1' to the 9-bit fractional input. This result is then conditionally right shifted by one bit if the sign bit indicates a negative input. This stage computes the base expression $1.\{x\}$ and adjusts it for the range $[-1,1)$.

Next, the integer part is evaluated to determine whether it is non-zero. Using two's complement representation, this occurs when the 2-bit integer is neither '00' (for [0, 1)) nor '11' (for $[-1, 0)$). This check is performed by applying a logical XOR to the integer part. If the integer part is non-zero, the output from the previous stage is shifted by one bit left/right for positive/negative input, effectively scaling the result by $2^1/2^{-1}$. Finally, the output multiplexer selects between the shifted and unshifted results to produce the final approximation of 2^x. As illustrated in Fig. 6, this architecture avoids multipliers and lookup tables, making it highly efficient for FPGA implementation while maintaining acceptable accuracy compared to conventional methods [16].

TABLE III
HARDWARE IMPLEMENTATION RESOURCES, POWER CONSUMPTION AND PERFORMANCE COMPARISON

Model	LUTs	FFs	Fmax (MHz)	Power (mW)	NRMSE (%)	Latency (ns)
[5] (Virtex-2)	1085	526	-	-	-	-
[6] (Virtex-2)	543	664	-	-	3.42	-
[7] (Virtex-2)	912	457	175	-	1.25	-
[8] (Kintex-7)	306	226	367.78	-	0.1404	87
[9] (Virtex-4)	161	216	320	295.49	1.43	37.5
[10] (Zynq)	375	221	450	-	0.00483	13.3
Proposed (Zynq)	140	141	224.56	26	0.36	13.36
	141	168	255.55	29	0.36	15.65
	146	235	357.14	46	0.36	16.8

C. Experimental Results

This section details the hardware implementation results of the proposed FHN neuron model, synthesized and deployed on the Xilinx Zynq FPGA. To highlight the capability of the proposed model to operate at various frequencies and balance latency and resource utilization, three pipeline configurations with different depths are implemented. Table III presents the results of the proposed model and previous FHN implementations. Latency, calculated as cycle count $\times$ clock period, for [8] and [9] was estimated from the hardware details and frequencies, while for [10], the 6-cycle latency was calculated based on its reported 6-stage register structure and clock frequency. Compared to the hardware-efficient design in [9], the proposed 4-stage version improves both hardware efficiency and computational performance, reducing the utilization of slice flip-flops and look-up tables by 22.2%and 12.4%, respectively, while enhancing accuracy by 74.8%. The architecture also achieves a latency of 15.65 ns, compared to the estimated 37.5 ns for the configuration in [9], based on its reported operating frequency and processing pipeline, which represents a 2.39$\times$ improvement. To ensure a fair comparison, power consumption was evaluated at a similar frequency (320 MHz), where the proposed model reduces dynamic power by 85.8%, consuming only 42 mW. Although the 4-stage version operates at a slightly higher frequency than the 3-stage design, its overall latency is higher because the added pipeline stage increases the cycle count without sufficient frequency gain. Notably, the 3-stage version offers latency comparable to [10] while requiring fewer hardware resources. Certain entries, marked with a hyphen (-) in Table III, indicate that the corresponding metrics were not reported in the cited works.

VI. CONCLUSION

In conclusion, this work presented an optimized hardware implementation of the FHN neuron model. By replacing the nonlinear term of the neuron equation with a power-of-two-based function, we proposed a hardware-friendly solution that enhances energy efficiency. Compared to existing implementations, our design achieves notable reductions in both resource usage and power consumption, while maintaining high computational accuracy. The combination of reduced resource usage, power consumption, and lower latency highlights the effectiveness of our approach for practical, low-power, real-time neuromorphic computing applications, where hardware constraints and energy efficiency are critical. These results demonstrate that the proposed model is well-suited for integration into larger neuromorphic systems requiring scalable, low-power neural components.

REFERENCES

[1] W. Gerstner and W.M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge, U.K.: Cambridge Univ. Press, 2002.

[2] A.L. Hodgkin and A.F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *J. Physiol.*, vol. 117, no. 4, pp. 500–544, Aug. 1952, doi: 10.1113/jphysiol.1952.sp004764.

[3] R. FitzHugh, "Impulses and physiological states in theoretical models of nerve membrane," *Biophys. J.*, vol. 1, no. 6, pp. 445–466, Jul. 1961, doi: 10.1016/s0006-3495(61)86902-6.

[4] J. Nagumo, S. Arimoto, and S. Yoshizawa, "An active pulse transmission line simulating nerve axon," *Proc. IRE*, vol. 50, no. 10, pp. 2061–2070, Oct. 1962, doi: 10.1109/jrproc.1962.288235.

[5] M. Nouri, G.R. Karimi, A. Ahmadi, and D. Abbott, "Digital multiplierless implementation of the biological FitzHugh–Nagumo model," *Neurocomputing*, vol. 165, pp. 468–476, Oct. 2015, doi: 10.1016/j.neucom.2015.03.084.

[6] M. Hayati, M. Nouri, S. Haghiri, and D. Abbott, "A digital realization of astrocyte and neural glial interactions," *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 2, pp. 518–529, Apr. 2016, doi: 10.1109/TBCAS.2015.2450837.

[7] A. Zahedi, S. Haghiri, and M. Hayati, "Multiplierless digital implementation of time-varying FitzHugh–Nagumo model," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 7, pp. 2662–2670, Jul. 2019, doi: 10.1109/TCSI.2019.2899361.

[8] A.J. Leigh, M. Heidarpur, and M. Mirhassani, "A low-resource digital implementation of the fitzhugh-nagumo neuron," in *Proc. 17th Conf. Ph.D Res. Microelectron. Electron. (PRIME)*, Villasimius, SU, Italy, Jun. 2022, pp. 369–372, doi: 10.1109/PRIME55000.2022.9816797.

[9] S. Haghiri, S.I. Yahya, A. Rezaei, and A. Ahmadi, "Multiplierless implementation of fitz-hugh nagumo (FHN) modeling using CORDIC approach," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 8, no. 1, pp. 279–287, Feb. 2024, doi: 10.1109/TETCI.2023.3300176.

[10] Y. Zuo et al., "Floating-point approximation enabling cost-effective and high-precision digital implementation of FitzHugh-nagumo neural networks," *IEEE Trans. Biomed. Circuits Syst.*, vol. 18, no. 2, pp. 347–360, Apr. 2024, doi: 10.1109/TBCAS.2023.3327496.

[11] S. Majidifar, M. Hayati, M.R. Malekshahi, and D. Abbott, "Low cost digital implementation of hybrid FitzHugh Nagumo–Morris lecar neuron model considering electromagnetic flux coupling," *IEEE Trans. Biomed. Circuits Syst.*, vol. 16, no. 6, pp. 1366–1374, Dec. 2022, doi: 10.1109/TBCAS.2022.3214851.

[12] S. Majidifar, M. Hayati, and S. Haghiri, "Exploring hybrid FitzHugh-rinzel (FHR) neuron model behavior: Cost-effective FPGA implementation for high-frequency and high-precision matching by electromagnetic flux effects," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 72, no. 2, pp. 844–853, Feb. 2025, doi: 10.1109/TCSI.2024.3503421.

[13] A. Carpegna, A. Savino, and S. Di Carlo, "Spiker: An FPGA-optimized hardware accelerator for spiking neural networks," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2022, pp. 14–19, doi: 10.1109/ISVLSI54635.2022.00016.

[14] L. Sayadi, A. Amirany, M.H. Moaiyeri, and S. Timarchi, "Balancing precision and efficiency: An approximate multiplier with built-in error compensation for error-resilient applications," *J. Supercomput.*, vol. 81, no. 1, p. 109, Nov. 2024, doi: 10.1007/s11227-024-06563-2.

[15] L. Sayadi, S. Timarchi, and A. Sheikh-Akbari, "Two efficient approximate unsigned multipliers by developing new configuration for approximate 4:2 compressors," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 4, pp. 1649–1659, Apr. 2023, doi: 10.1109/TCSI.2023.3242558.

[16] J. Pu, W.L. Goh, V.P. Nambiar, Y.S. Chong, and A.T. Do, "A low-cost high-throughput digital design of biorealistic spiking neuron," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 4, pp. 1398–1402, Apr. 2021, doi: 10.1109/TCSII.2020.3023825.