2024 International Conference on Intelligent and Innovative Computing Applications (ICONIC)
7th-9th November 2024, Pearle Beach Resort & Spa, Flic en Flac, Mauritius
https://doi.org/10.59200/ICONIC.2024.027

Spiking Neural Network Models Analysis on Field Programmable Gate Arrays

Shamini Koravuna[1] and Ulrich Rückert[4]
Bielefeld University
Bielefeld, Germany
skoravuna@techfak.uni-bielefeld.de, rueckert@techfak.uni-bielefeld.de

Sanaullah[2] and Thorsten Jungeblut[3]
Bielefeld University of Applied Sciences and Arts
Bielefeld, Germany
sanaullah@hsbi.de, thorsten.jungeblut@hsbi.de

Abstract
Spiking Neural Networks (SNNs) are a type of neural network designed to replicate biological neural networks more closely by using discrete spikes to transmit information. Unlike traditional networks, SNNs incorporate time by relying on the precise timing of spikes for neuron-to-neuron communication. This reduces hardware complexity, as it only requires one-bit logic, making SNNs ideal for hardware integration. This study assesses the performance of several SNN models for hardware implementation, focusing on resource utilization, speed, and power consumption. Verilog was used for the hardware design, and the simulations were run in Vivado. The emulation experiments were conducted on the Basys3 FPGA board to validate our findings. Our analysis indicates that simpler models like Leaky Integrate and Fire (LIF) and Non-linear Integrate-and-Fire (NLIF) are highly efficient, with low resource and power requirements, making them suitable for resource-constrained environments. More complex models like Hodgkin-Huxley (HH) and Izhikevich (IZH) provide detailed neuronal dynamics but at a higher resource cost. Our implementations exhibit notable improvements across several metrics compared to previous work. This analysis equips researchers with the necessary information to make informed decisions about which neuron model best meets their application needs, whether prioritizing speed, efficiency, or biological accuracy.

Keywords: Spiking Neural Networks, Neuron Models, FPGAs, Verilog

1. Introduction

Spiking Neural Networks (SNNs) are designed to mimic the biological nervous system. In the brain, neurons communicate by sending trains of action potentials, known as spike trains. SNNs imitate this mechanism, so a neuron is only activated when a new input spike arrives. Additionally, SNNs are sensitive to the temporal characteristics of information transmission, which is an advantage over other neural networks (Maass, 1997). However, the corresponding computation requirements increase as the SNN model becomes more complex. Customized hardware accelerators are required to achieve higher computing/power efficiency, especially for embedded and lightweight applications.

SNNs are typically run on CPUs using software frameworks like Nest (Eppler et al. 2009), Brian (Stimberg et al. 2000), RAVSim (Sanaullah et al. 2022), or on GPUs with frameworks like NEMO (Fidjeland et al. 2010), CNS (Mutch et al. 2010), NCS6 (Hoang et al. 2013), CARLsim (Niedermeier et al. 2022). However, these hardware platforms often consume a lot of power due to limited memory bandwidth, which also reduces their execution speed. To overcome these limitations, many researchers have developed custom ASICs (Application-Specific Integrated Circuits), such as IBM's TrueNorth (Akopyan et al. 2015) and SpiNNaker (Furber et al. 2020), to boost performance and energy efficiency. Despite their advantages, ASICs can be inflexible, especially as neural network models evolve and new layers are introduced. Additionally, designing and producing large ASIC chips is both expensive and time-consuming, making it difficult to keep pace with rapid advancements in

the field. In situations where hardware resources are limited, FPGAs (Field-Programmable Gate Arrays) offer a flexible alternative, allowing different accelerators to be implemented on the same hardware block (Neil et al. 2014) (Wang et al. 2017) (Carpegna et al. 2022). However, the complexity of FPGA designs can make it challenging to deploy large networks, particularly in edge computing scenarios.

This study evaluates the suitability of nine popular SNN models for hardware implementation. The models under examination are: Leaky Integrate-and-Fire (LIF) (Gerstein et al. 1964), Non-linear Integrate-and-Fire (NLIF) (Jolivet et al. 2004), Integrate-and-Fire with Spike Frequency Adaptation (IF-SFA) (Gigante et al. 2007), Quadratic Integrate-and-Fire (QIF) (Brunel et al. 2003), Adaptive Exponential (AdEx) (Brette et al. 2005), Spike Response Model (SRM) (Gerstner 2008), Theta Model (McKennoch et al. 2009), Hodgkin-Huxley (HH) (Häusser 2000), and Izhikevich Model (IZH) (Izhikevich 2004). Our evaluation is based on a comprehensive study by Sanaullah et al. (2023), which simulated these neuronal models on a CPU. We assessed each model in terms of resource utilization, operational speed, and power consumption. For the implementation and simulation of these SNN models, we utilized the Xilinx Vivado simulator and Verilog Hardware Description Language (HDL). Additionally, we conducted emulation experiments on the Basys3 FPGA development board to validate our findings. The remainder of the paper is organized as follows: Section 2 reviews related works in the field of SNN implementations on FPGAs. Section 3 details the SNN models. Section 4 provides the hardware implementation. Section 5 provides a comprehensive evaluation of the models. Finally, Section 6 concludes the paper.

2. Related works

Neurons in the human brain vary in several ways, including the type of neurotransmitters they use, their morphological characteristics, and their spiking patterns. Despite these differences, they share a fundamental structure consisting of three main components: a dendritic tree (input channels), an axon (output mechanism), and a soma (core) (Gerstner et al. 2014). The dendrites and axons can be further divided into multiple segments through which signals travel, allowing for complex processing and communication across neural networks. Inspired by this biological structure of the neurons, neuromorphic hardware implementations of SNNs aim to replicate these dynamics. The LIF neuron is one of the simplest and most used neuron models. It is often implemented using an RC-parallel circuit (Dutta et al. 2017). The model considers the membrane as a leaky capacitance with a resistive element that causes the voltage to move towards a resting value when there is no input stimulation. Additionally, various modifications to the LIF model have been explored, such as the AdEx neuron model proposed by Heidarpour et al. (2016). This model integrates the potential difference exponentially in the current equation, mimicking the non-linear increase in action potential observed in biological neurons. Despite the higher implementation cost of non-linear behavior on hardware, the authors discretized the differential equations using the Euler method and implemented them on a Xilinx Spartan 6 FPGA using simple add and shift operations only. Similarly, Basham et al. (2012) introduce a QIF neuron model, where the current equation integrates the square of the potential difference. This design utilizes a fixed-point multiplier to evaluate the square of the voltage.

Furthermore, Fang et al. (2019) proposed an FPGA-based SNN featuring biologically realistic neurons and synapses tailored for temporal information processing. Their model employs the LIF neuron and dynamic synapse model with three kernel types (Exponential, Alpha, and Dual exponential) alongside a hybrid encoding scheme combining population coding and temporal coding. Validation using the MNIST dataset reveals higher accuracy compared to previous works. A similar event-driven approach presented by Neil et al. (2014) for the IF model yielded lower accuracy, potentially attributed to overlooking synapse dynamics. Evaluation against CPUs, GPUs, and rate encoding using the Australian Sign Language dataset further demonstrates the superiority of the proposed hardware architecture and encoding scheme (Fang et al. 2019).

Kumar et al. (2016) developed a spiking neuron model of the HH on an FPGA, leveraging a hardware architecture built upon a series of adder and multiplier blocks to execute the first-order differential equations in Verilog on Xilinx Virtex-5. However, while representing each neuron parameter in fixed-point with chosen bit lengths, their justification for bit optimization strategies lacks consistency. The utilization of multiplier

blocks led to an increase in the number of Look-Up Tables (LUTs) and, consequently, increased power consumption. Conversely, Shama et al. (2020) proposed a multiplier-less HH model aiming to overcome the computational complexity of the HH model, which involves numerous complicated equations requiring many multiplications. Their approach approximated hyperbolic functions as piece-wise linear terms, executing all multiply operations as logical shifts and additions. This was made feasible by modifying equations to power-2-based functions. Despite the high accuracy of the HH neuron, a multiplier-less design further reduced operational costs and increased frequency. Notably, their model achieved a 69% reduction in FPGA resources and a maximum processing frequency of 85 MHz, implemented on a Virtex-2 FPGA with fewer hardware resources compared to works using high-performance platforms such as Virtex-7 and Artix-7. Bonabi et al. (2014) also presented a similar approach with a 58% reduction in FPGA resource usage.

The impact of timestep on the performance and accuracy of the Izhikevich model was investigated by Heidarpur et al. (2020), who proposed a combined approach of software simulation and hardware implementation. By increasing the timestep to a threshold value, they observed dumped oscillations leading to neuron instability under different input currents. They validated their software simulation results by implementing a neuron model on FPGA to ensure stability. Their discussion highlighted that smaller timesteps require more system resources. Additionally, Alkabaa et al. (2022) evaluated a model based on its complexity and hardware resources required for FPGA realization, focusing on the two-neuron coupled Izhikevich model. Utilizing a LUT-based approach for quadratic equations instead of approximation techniques, they achieved closer alignment with mathematical equations as the LUT size increased.

Another unique technique involving the Izhikevich neuron model is discussed by Karaca et al. (2021), where the authors explored its practical suitability for electrical realization due to its chaotic behavior. They compared a modified model version with the original coupled neuron dynamics through numerical simulations and FPGA demonstrations, noting a longer processing time but reduced utilization numbers without using multipliers. Furthermore, Niu et al. (2012) emulated motoneurons in the motor nervous system with the Izhikevich model on an FPGA to analyze pediatric neurological diseases, extending the application of these neuron models beyond. Koravuna et al. (2023) demonstrated a real-time FPGA implementation of an SNN for pattern recognition using IZH neurons. The hardware cost was reduced by implementing a multiplier-less approximation.

3. Neuronal Models

Neuronal models are essential tools in computational neuroscience, enabling researchers to simulate and analyze the complex dynamics of neural systems. SNNs are modeled using mathematical equations that describe the dynamics of spiking neurons, capturing parameters such as input current, membrane potential, and membrane time constants to mimic biological neurons' behavior. The choice of neuron model plays a significant role in determining the balance between computational efficiency and biological realism. Various SNN models utilize distinct mathematical frameworks to capture different neural characteristics, offering diverse perspectives on neural dynamics.

This study investigates several key SNN models, including the LIF, NLIF, IF-SFA, QIF, AdEx, SRM, Theta, HH, and IZH. Each model represents a unique approach to simulating neuron behavior, enabling a comprehensive analysis of spiking neuron dynamics.

3.1 LIF

The LIF model is simple, yet it effectively represents neuron dynamics, capturing the membrane potential's decay and input integration over time. The membrane potential dynamics is:

$$\tau \frac{dV(t)}{dt} = -(V(t) - V_r) + (R_m I(t))/g_l \qquad (1)$$

Where:

- $V(t)$ is the membrane potential at time t.
- V_r is the resting potential.
- $\tau = R_m . C_m$ is the membrane time constant.
- R_m is the membrane resistance.
- C_m is the membrane capacitance.
- $I(t)$ is the input current.
- g_l is the leak conductance.

When the membrane potential reaches the threshold potential V_{th} (i.e., $V(t) \geq V_{th}$), a spike is emitted, and the membrane potential is reset to

the resting potential V_{reset}. After the spike, the neuron enters a refractory period t_{ref}, during which it is prevented from integrating input.

3.2 NLIF

The NLIF model is slightly more complex than the LIF neuron model. The mathematical equation is:

$$\tau \frac{dV(t)}{dt} = -(V(t) - V_r) + (R_m I(t))/g_l + V(t)^2 \quad (2)$$

The additional $V(t)^2$ term introduces nonlinearity, capturing more complex membrane behavior. When $V(t) \geq V_{th}$, a spike occurs, and $V(t)$ is reset, and the neuron enters the refractory period.

3.3 IF-SFA

The IF-SFA model captures spike-frequency adaptation, where the firing rate decreases over time when the neuron is stimulated with a constant current. This phenomenon is observed in many cortical neurons, making the IF-SFA model a good balance between simplicity and biological realism. Here is the mathematical representation:

$$\tau \frac{dV(t)}{dt} = -(V(t) - V_r) + \frac{R_m I(t)}{g_l} + \omega(t) \quad (3)$$

$$\tau_\omega \frac{d\omega(t)}{dt} = -\omega(t) + b.\delta(t - t_{spike}) \quad (4)$$

Where:

- $\omega(t)$ is the adaptation current.
- τ_ω is the adaptation time constant.
- $\delta(t - t_{spike})$ represents the spike-triggered adaptation, where t_{spike} is the time of a spike.
- b is the adaptation strength.

When $V(t) \geq V_{th}$, a spike occurs, and $V(t)$ is reset. Also, the adaptation current is updated as $\omega(t) = \omega(t) + b$.

3.4 QIF

The QIF model is nonlinear but can still be simulated efficiently compared to more complex models. It provides a more accurate description near the spike onset. It is represented as:

$$\tau \frac{dV(t)}{dt} = -(V(t) - V_r) + \sqrt{C_m}\sqrt{I(t)} \quad (5)$$

The model incorporates a quadratic voltage term near the firing threshold. When $V(t) \geq V_{th,}$ a spike is emitted, followed by a refractory period, and the membrane potential remains at the reset potential.

3.5 AdEx

The AdEx model combines adaptive dynamics and nonlinear spike generation, providing realistic neuron behavior, including spike-frequency adaptation. The mathematical equations are:

$$\tau \frac{dV(t)}{dt} = -(V(t) - V_r) + I(t) + \omega(t) + \Delta_T \exp\left(\frac{V(t) - V_{th}}{\Delta_T}\right) \quad (6)$$

$$\tau_\omega \frac{d\omega(t)}{dt} = aV(t) - \omega(t) \quad (7)$$

Where, a represents the subthreshold adaptation conductance and Δ_T is the slope factor. When $V(t) \geq V_{th}$, a spike occurs, V and ω are updated similarly to IF-SFA. AdEx strikes a balance between simplicity and biological realism, offering more accurate adaptation dynamics.

3.6 SRM

SRM models are efficient and capture both synaptic dynamics and spike-based interactions. It is realized with these differential equations:

$$\tau \frac{dV(t)}{dt} = -V(t) + I(t) + \eta(t) + \epsilon(t) \quad (8)$$

$$\eta(t) = \eta_0 exp^{-\frac{t}{\tau_\eta}} \quad (9)$$

$$\epsilon(t) = \epsilon_0 exp^{-\frac{t}{\tau_\epsilon}} \quad (10)$$

Where:

- $\eta(t)$ is the response to the neuron's past spikes.
- $\epsilon(t)$ is the postsynaptic response to incoming spikes from other neurons.
- η_0 is the initial amplitude of the refractory effect.
- τ_η is the time constant that controls the refractory kernel's decay rate.
- ϵ_0 is the initial amplitude of the postsynaptic response.
- τ_ϵ is the time constant for postsynaptic decay.

When $V(t) \geq V_{th}$, a spike occurs, and $V(t)$ is reset, the refractory kernel is updated as $\eta(t) = \eta(t) + \eta_{spike}$, where η_{spike} is the spike-triggered

increment added to the kernel immediately after the spike, and the synaptic kernel is updated as $\epsilon(t) = \epsilon(t) + \epsilon_{spike}$, where ϵ_{spike} is the magnitude of the postsynaptic response to the incoming spike.

3.7. Theta

The Theta model is commonly used in phase-oscillator models and captures rhythmic spiking activity. The model is ideal for analyzing neuronal phase dynamics and synchronization. The mathematical representation is:

$$\tau \frac{dV(t)}{dt} = -(V(t) - V_r) + g\big(\theta - V(t)\big) + I(t) \quad (11)$$

Where, g is the input conductance and θ is the threshold. When $V(t) \geq \theta$, a spike occurs, and $V(t)$ is reset.

3.8 HH

Based on the original experiments with the squid axon [Schwiening 2012], the HH model is one of the most biologically accurate descriptions of action potential generation. The differential equations are:

$$C_m \frac{dV(t)}{dt} = I(t) + I_{Na} + I_K + I_l \quad (12)$$

$$I_{Na} = g_{Na} m^3 h (V(t) - E_{Na}) \quad (13)$$

$$I_K = g_K n^4 (V(t) - E_K) \quad (14)$$

$$I_l = g_l (V(t) - E_l) \quad (15)$$

$$\frac{dn}{dt} = \alpha_n V(t)(1 - n) - \beta_n V(t) n \quad (16)$$

$$\frac{dm}{dt} = \alpha_m V(t)(1 - m) - \beta_m V(t) m \quad (17)$$

$$\frac{dh}{dt} = \alpha_h V(t)(1 - h) - \beta_h V(t) h \quad (18)$$

Where:

- I_{Na}, I_K & I_l are Sodium, Potassium, and leak currents, respectively.
- g_{Na}, g_K & g_l represent the maximum conductance of the ion channels.
- $m, n \; and \; h$ are the gating variables.
- $E_{Na}, E_K \; and \; E_l$ are the reversal potentials of the ion channels.
- $\alpha_n(V(t))$, $\alpha_m(V(t)), \alpha_h(V(t)), \beta_n(V(t))$, $\beta_m(V(t))$, $and \; \beta_h(V(t)$ represent the rate functions to control the opening and closing of ion channels.

When $V(t) \geq V_{th}$, a spike occurs, and $V(t)$ is reset, and the gating variables $m, n \; and \; h$ are also updated based on the rate functions. The model is computationally expensive because it requires solving multiple nonlinear differential equations.

3.9 IZH

The Izhikevich model is known for its flexibility, which is capable of reproducing various firing patterns observed in biological neurons, from regular spiking to bursting. The model balances biological realism with computational efficiency, suitable for large network simulations. The neuronal dynamics are given as follows:

$$\frac{dV(t)}{dt} = 0.04V(t)^2 + 5V(t) + 140 - U(t) + I(t) \quad (19)$$

$$\frac{dU(t)}{dt} = a\big(bV(t) - U(t)\big) \quad (20)$$

Where:

- $U(t)$ is the recovery variable, which provides adaptation.
- $a, and \; b$ are model parameters.

When $V(t) \geq 30\, mV$, a spike is emitted, and $V(t) and \; U(t)$ are reset: $V(t) = c, U(t) = U(t) + d$, where $c \; and \; d$ are dimensionless variables.

Understanding the variety of neuron models available is crucial for selecting the appropriate model depending on the research objectives and computational constraints. The models covered in this paper—from the LIF and NLIF models to more complex frameworks like the HH model—represent key milestones in neural modeling. Each model offers distinct trade-offs between simplicity, biological accuracy, and computational demands.

4. Hardware Implementation

This study aims to implement and emulate biologically inspired neuronal models on an FPGA, taking advantage of the parallelism and speed offered by hardware design. The model designs were synthesized for the Basys3 FPGA,

which uses a Xilinx Artix-7 device. The FPGA implementation allows real-time emulation of neurons, making it suitable for large-scale parallel simulation.

Several design considerations were made to efficiently implement the hardware-friendly neuronal models. The neuronal models use fixed-point arithmetic to reduce hardware complexity and resource utilization. This choice allows the FPGA to compute neuron dynamics efficiently without requiring floating-point units. The design uses a finite state machine (FSM) to manage the different states of the neuron (e.g., idle and firing). This approach simplifies the control logic and ensures accurate spike generation. The model parameters, such as membrane resistance, threshold potential, and time constants, are all configurable using Verilog parameters. This flexibility makes it easy to adapt the model to different biological scenarios. The designs were optimized by minimizing the use of multipliers and divisions and leveraging shift operations for the multiplication and division operations, as these operations in digital design can be particularly costly in terms of timing and resource usage.

Adapted Models

LIF

The design parameters are adapted as follows:

$$\tau = 2^{\tau_s}, R_m = 2^{R_{ms}}, and\ g_l = 2^{g_{ls}}$$

The approximated mathematical model for FPGA implementation:

$$V(t) = (-V(t-1) + V_r + (I(t) \ll R_{ms}) \gg g_{ls}) \gg \tau_s \quad (21)$$

NLIF

The design parameters are adjusted as follows:

$$\tau = 2^{\tau_s}, R_m = 2^{R_{ms}}, and\ g_l = 2^{g_{ls}}$$

The mathematical model realized is:

$$V(t) = (-V(t-1) + V_r + (I(t) \ll R_{ms}) \gg g_{ls} + V(t-1)^2) \gg \tau_s \quad (22)$$

IF-SFA

The design specifications are modeled as follows:

$$\tau = 2^{\tau_s}, \tau_\omega = 2^{\tau_{\omega s}}, R_m = 2^{R_{ms}}, b = 2^{bs}\ and\ g_l = 2^{g_{ls}}$$

The derived mathematical model for hardware implementation is:

$$V(t) = \left(-V(t-1) + V_r + (I(t) \ll R_{ms}) \gg g_{ls} + \omega(t)\right) \gg \tau_s \quad (23)$$

$$\omega(t) = \left(-\omega(t-1) + \delta\left(t - t_{spike}\right) \ll bs\right) \gg \tau_{\omega s} \quad (24)$$

QIF

The design constraints have been redefined as follows:

$$\tau = 2^{\tau_s},\ and\ C_m = 2^{C_{ms}}$$

The simplified mathematical model for FPGA deployment is:

$$V(t) = (-V(t-1)^2 + V_r + (I(t) \ll C_{ms})) \gg \tau_s \quad (25)$$

AdEx

The design parameters are reconfigured as:

$$\tau = 2^{\tau_s},\ \tau_\omega = 2^{\tau_{\omega s}}, a = 2^{a_s}, and\ \Delta_T = 2^{\Delta_{Ts}}$$

The revised mathematical model is:

$$V(t) = (-V(t-1) + V_r + I(t) + \omega(t) + (((V(t-1) - V_{th}) \ll \Delta_{Ts}) + 1) \ll \Delta_{Ts}) \gg \tau_s \quad (26)$$

$$\omega(t) = \left(V(t-1) \ll a_s - \omega(t-1)\right) \gg \tau_{\omega s} \quad (27)$$

SRM

The following updates have been made to the design parameters:

$$\tau = 2^{\tau_s}, \tau_\eta = 2^{\eta_s},\ and\ \tau_\epsilon = 2^{\epsilon_s}$$

The optimized mathematical model is:

$$V(t) = (-V(t-1) + I(t) + \eta(t) + \epsilon(t)) \gg \tau_s \quad (28)$$

$$\eta(t) = \eta(t-1) - (\eta(t-1) \gg \eta_s) \quad (29)$$

$$\epsilon(t) = \epsilon(t-1) - (\epsilon(t-1) \gg \epsilon_s) \quad (30)$$

Theta

The design parameters are updated as follows:

$$\tau = 2^{\tau_s}\ and\ g = 2^{g_s}$$

The computed mathematical model is:

$$V(t) = \Big(-(V(t-1) - V_r) + \big(\theta - V(t-1)\big) \ll g_s + I(t)\Big) \gg \tau_s \quad (31)$$

HH

The design parameters are adapted as follows:

$C_m = 2^{C_{ms}}, \quad g_{Na} = 2^{g_{Na\_s}}, \quad g_K = 2^{g_{K\_s}}, \; and \; g_l = 2^{g_{l\_s}}$

The approximated mathematical model is:

$$V(t) = (I(t) + I_{Na} + I_k + I_l) \gg C_{ms} \quad (32)$$

$$I_{Na} = (m^3(V(t-1) - E_{Na})h) \ll g_{Na\_s} \quad (33)$$

$$I_{Na} = \big(n^4(V(t-1) - E_K)\big) \ll g_{K\_s} \quad (34)$$

$$I_l = (V(t-1) - E_l) \ll g_{l\_s} \quad (35)$$

IZH

The design parameters are set as follows:

$$a = 2^{a_s} \; and \; b = 2^{b_s}$$

The simplified mathematical model for hardware implementation is:

$$V(t) = 2V(t) + 4V(t) + (V(t)^2 \gg 4) + 140 - U(t) + I(t) \quad (36)$$

$$U(t) = \big((V(t) \ll a_s) - U(t)\big) \ll b_s \quad (37)$$

5. Results and Discussion

This section presents and analyzes the results from the hardware emulation of various neuronal models on the Basys3 FPGA platform. We begin by discussing the neuronal parameters used and their spiking activities. Then, we evaluate the models based on key metrics such as resource utilization, power consumption, and performance, focusing on their operating frequencies.

5.1 Spiking Activity

The spiking activities of different neuronal models were visualized over time during the simulation. Figure 1 illustrates the spiking patterns of each model in response to an input current pulse. The parameter values used for this comparison are listed to highlight the differences in spiking behavior across the models. A constant input current was applied simultaneously to all models. As the input is integrated by each model, the membrane potentials increase. When a model's membrane potential reaches a predefined threshold, it generates a spike and the potential resets to its resting value. After each spike, a brief refractory period occurs, during which the neuron is temporarily inactive before it can spike again.

The spiking patterns, response times, and spike counts vary between models due to their unique dynamics and parameter configurations. The comparison in Figure 1 demonstrates how each neuron model responds to the same input current (I(t) = 50), revealing significant differences in their behavior.

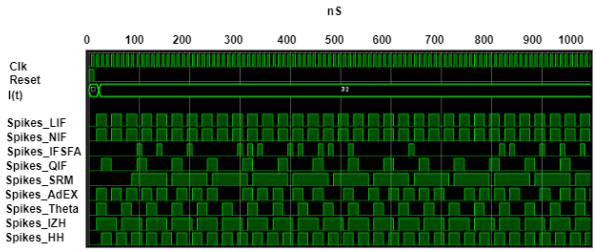

Figure 1: Spiking activity of different SNN models in response to an input current pulse.

Parameters:

LIF: $\tau = 8, \quad V_{th} = -64, \quad R_m = 8, \quad g_l = 8, \; V0 = -64$ (initial potential), $V_{reset} = -70, \; V_r = 0.$

NLIF: $\tau = 8, \quad V_{th} = -64, \quad R_m = 8, \quad g_l = 8, \; V0 = -64$, $V_{reset} = -70, \; V_r = 0.$

IF-SFA: $\tau = 8, \; V_{th} = -64, \; R_m = 8, \; g_l = 8, \; V0 = -64, \; V_{reset} = -70, \; b = 8, \; \tau_\omega = 8.$

QIF: $\tau = 8, \; V_{th} = -64, \; C_m = 8, \; V0 = -64, \; V_{reset} = -70.$

AdEx: $\tau = 8, \; V_{th} = -64, \; \tau_\omega = 8$, $a = 4, \; V0 = -64, \; V_{reset} = -70, \; \Delta_T = 8.$

SRM: $\tau = 3, \quad V_{th} = -64, \quad \epsilon(t) = 8$, $\epsilon_{spike} = 5, \; V0 = -64, \quad V_{reset} = -70, \; \eta(t) = 8, \; \eta_{spike} = 10.$

Theta: $\tau = 8, \; V_{th} = -64, \; V0 = -64, \quad g = 8, \; V_{reset} = -70.$

HH: $V_{th} = 50, \; g_l = 3, \; V_{reset} = 0, \quad C_m = 10, \; g_{Na} = 120, \; g_K = 36, \quad E_K = -82, \quad E_l = -84, \; E_{Na} = 50.$

IZH: $V_{th} = 30, \quad a = 2, \; b = 16, \; c = -65, \; d = 8, V0 = 0.$

5.2 Performance Analysis

Table 1 shows the FPGA's performance analysis based on resource usage in terms of LUTs, Flip-Flops (FFs), and DSP blocks for each neuronal

model, the power consumption and maximum operational frequency.

Resource Utilization

The resource utilization results show that the **LIF and NLIF** models exhibit the most efficient use of hardware resources, with 13 and 12 LUTs, respectively, and both require only 17 FFs. This minimal use of resources reflects the simplicity of these models, which focus on basic integration and threshold dynamics without complex biological features. The **IF-SFA** model, which introduces adaptation mechanisms into the basic IF model, requires comparatively more resources, utilizing 38 LUTs and 30 FFs. The adaptation mechanism likely accounts for this increase in complexity, as the model needs additional logic to adjust the neuron's response to previous firing activity. This makes the IF-SFA more biologically realistic but with a noticeable rise in hardware cost compared to the basic LIF and NLIF. Similarly, the **Theta** model shows moderate hardware resource utilization offering a good middle ground between biological realism and computational efficiency

The **QIF** model, designed to capture more intricate neuronal firing patterns by incorporating quadratic terms, is among one of the most resource-heavy, utilizing 82 LUTs and 21 FFs. This indicates a significant jump in complexity as compared to simpler models. Similarly, **AdEx** (48 LUTs and 29 FFs), and **SRM** (65 LUTs and 45 FFs), represent biologically detailed models that aim to capture various neuron dynamics. For instance, the **AdEx** model introduces exponential spike generation, while **SRM** models the neuron's response to input spikes based on synaptic kernels.

DSP Blocks are only used by two neuron models. **HH** consumes 3 DSP blocks, and the **IZH** model uses 1 DSP block. DSP blocks are typically employed to perform complex arithmetic operations efficiently. The **HH** model, known for simulating ion channel dynamics using differential equations, requires dedicated DSP resources for the continuous, time-dependent processes involved. The **IZH** model, while offering a good trade-off between biological accuracy and computational simplicity, requires some DSP support, likely due to its ability to simulate both regular spiking and bursting patterns using more advanced mathematical functions.

Table 1: FPGA performance analysis

Neuron	LUT	FF	DSP	Power (W)	Max. Frequency (MHz)
LIF	13	17	-	0.092	450
NLIF	12	17	-	0.096	450
IF-SFA	38	30	-	0.103	270
QIF	82	21	-	0.085	148
AdEx	48	29	-	0.093	189
SRM	65	45	-	0.099	239
Theta	30	21	-	0.099	236
HH	73	50	3	0.105	115
IZH	42	25	1	0.099	130

Power Consumption

Power consumption is an important metric, especially in applications where energy efficiency is critical, such as mobile or embedded neuromorphic systems. Across all models, the power consumption is relatively consistent, ranging from 0.085 W to 0.105 W. **LIF and NLIF** have moderate power consumption at around 0.092 W and 0.096 W, respectively balancing simplicity with computational needs. This slight increase compared to **QIF** is due to their higher operating frequencies. The models, such as **AdEx**, **SRM**, **Theta**, and **Izhikevich**, consume between 0.093 W to 0.105 W, reflecting their more complex dynamics that require additional logic to maintain, particularly in cases where exponential terms, adaptation, or other biologically plausible features are simulated. The **IF-SFA** model consumes 0.103 W of power; this increase in power usage is due to the additional logic required for the adaptation mechanism, which tracks the neuron's firing history and adjusts the membrane potential accordingly. The **HH** model, at 0.105 W, consumes the most power, reflecting its computational complexity and the requirement for DSP blocks.

Operating Frequency

Maximum operating frequency is another critical performance parameter, especially in time-sensitive neuromorphic applications that demand fast real-time processing. The **LIF** and **NLIF** models achieve the highest maximum frequency, operating at 450 MHz, which makes them ideal for high-speed, low-latency neural simulations. This high performance can be attributed to their simplicity, which allows for faster propagation of

signals through the hardware without significant delays caused by complex operations. Models like **IF-SFA** (270 MHz), **AdEx** (189 MHz), **SRM** (239 MHz), and **Theta** (236 MHz) achieve relatively high frequencies, although lower than LIF and NLIF. These intermediate operating frequencies reflect the additional processing required for spike-frequency adaptation, synaptic kernels, or more detailed voltage dynamics. Despite these complexities, they remain suitable for real-time applications where moderately high-speed processing is sufficient. On the lower end, models such as **QIF** (148 MHz), **HH** (115 MHz), and **Izhikevich** (130 MHz) operate at much lower frequencies, with the **HH** model being the slowest. This significant drop in frequency is due to the increased computational load from more complex mathematical operations and the continuous nature of the neuronal dynamics. These models may struggle in scenarios requiring high-speed, real-time simulations but offer much greater biological realism, making them more suitable for tasks where accuracy is prioritized over speed.

Table 2 shows the performance metrics of previous work for the available implementations on FPGAs. Our implementations exhibit notable improvements across several metrics compared to previous work. Our LIF model, with 13 LUTs and 17 FFs, consumes 0.092 W and achieves a maximum frequency of 450 MHz, significantly higher than Fan et al. (2022), who used 593 LUTs, 63 FFs, and achieved 83 MHz at 0.142 W. Similarly, our QIF model, using 82 LUTs and 21 FFs, consumes 0.085 W and operates at 148 MHz, significantly improving upon the 86 LUTs and 41 FFs reported by Bashram et al. (2012), which only achieved 12 MHz. The AdEx model, with 48 LUTs and 29 FFs, achieves 189 MHz while consuming 0.093 W, compared to the 472 LUTs and 185 FFs of Wang et al. (2022) at 212 MHz. Our IZH model uses 42 LUTs and 25 FFs, consumes 0.099 W, and runs at 130 MHz, an improvement in resource efficiency compared to Yang et al. (2020), who used 119 LUTs and 130 FFs but with a lower frequency of 291.8 MHz. These improvements demonstrate the efficiency and scalability of our models compared to prior work.

Table 2: Performance metrics of previous work

Work Neuron	**Resources (LUTs, FFs, DSPs) FPGA**	**Power (W)**	**Max Frequency (MHz)**
Carpegna et al. (2022) LIF	62, 40, - Artix-7	-	-
Fan et al. (2022) LIF	593, 63, - Cyclone-4	0.142	83
Basham et al. (2012) QIF	84, 41, - Spartan 3	0.034	12
Muñoz et al. (2012) SRM	378, 202, - Spartan 3	-	-
Wang et al. (2022) AdEX	472, 185, - Virtex II	-	212
Yang et al. (2020) IZH	119, 130, - ZCU102	-	291.8
Our Best Models – Basys3- Artix 7			
LIF	13, 17, -	0.092	450
NLIF	12, 17, -	0.096	450

6. Conclusion

This study focused on implementing and evaluating various neuronal models on FPGA to assess their suitability for real-time hardware emulation. We successfully deployed and tested the LIF, NLIF, IF-SFA, QIF, AdEx, SRM, Theta, HH, and IZH models on the Basys3 FPGA board, examining their resource utilization, power consumption, and their maximum operating frequency. Our results demonstrate that simpler models like LIF and NLIF offer high efficiency in terms of hardware resource utilization and power consumption while achieving the highest operating frequencies. Their minimal resource requirements make them highly suitable for real-time, high-speed applications where large-scale simulations or low-latency processing are critical. Intermediate complexity models, including IF-SFA, AdEx, SRM, and Theta, offer a balanced approach between biological realism and operational efficiency. These models require more resources and exhibit lower frequencies than simpler models, but they provide enhanced dynamical features and adaptation mechanisms, making them appropriate for applications where a

compromise between accuracy and performance is acceptable. Complex neuron models such as QIF, HH, and IZH, although more resource-intensive and slower, offer significant improvements in simulating neuronal behavior. Specifically, HH and IZH are more biologically realistic models. These models, with their higher computational demands and lower frequencies, are best suited for applications where the accuracy of biological dynamics is paramount, such as detailed neural simulations or advanced brain modeling. Overall, this work contributes valuable insights into the trade-offs between computational complexity, resource utilization, and performance in FPGA-based neuronal emulations.

Acknowledgments
This research was supported by the research training group "Dataninja" (Trustworthy AI for Seamless Problem Solving: Next-Generation Intelligence Joins Robust Data Analysis), funded by the German federal state of North Rhine-Westphalia.

7. References

Akopyan, F., Sawada, J., Cassidy, A.S., Alvarez-Icaza, R., Arthur, J.V., Merolla, P., Imam, N., Nakamura, Y., Datta, P., Nam, G., Taba, B., Beakes, M.P., Brezzo, B., Kuang, J.B., Manohar, R., Risk, W.P., Jackson, B.L., & Modha, D.S. (2015). TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *34*, 1537-1557. https://doi.org/10.1109/TCAD.2015.2474396

Alkabaa, A. S., Taylan, O., Yilmaz, M. T., Nazemi, E., & Kalmoun, E. M. (2022). An investigation on spiking neural networks based on the izhikevich neuronal model: Spiking processing and hardware approach. *Mathematics*, *10*(4), 612. https://doi.org/10.3390/math10040612

Basham, E.J., & Parent, D.W. (2012). Compact digital implementation of a quadratic integrate-and-fire neuron. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, San Diego, CA, USA, 2012, pp. 3543-3548, doi: 10.1109/EMBC.2012.6346731.

Brette, R., & Gerstner, W. (2005). Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Journal of neurophysiology*, *94*(5), 3637-3642. https://doi.org/10.1152/jn.00686.2005

Brunel, N., & Latham, P. E. (2003). Firing rate of the noisy quadratic integrate-and-fire neuron. *Neural computation*, *15*(10), 2281-2306. https://doi.org/10.1162/089976603322362365

Carpegna, A., Savino, A., & Carlo, S.D. (2022). Spiker: an FPGA-optimized Hardware accelerator for Spiking Neural Networks. *2022 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 14-19. https://doi.ieeecomputersociety.org/10.1109/ISVLSI54635.2022.00016

Dutta, S., Kumar, V., Shukla, A., Mohapatra, N.R., & Ganguly, U. (2017). Leaky Integrate and Fire Neuron by Charge-Discharge Dynamics in Floating-Body MOSFET. *Scientific Reports*, *7*.

Eppler, J. M., Helias, M., Muller, E., Diesmann, M., & Gewaltig, M. O. (2009). PyNEST: a convenient interface to the NEST simulator. *Frontiers in neuroinformatics*, *2*, 363. https://doi.org/10.3389/neuro.11.012.2008

Fan, Y., Deng, D., Wang, J., & Yang, S (2022). Power-efficient and Multiplier-less FPGA Implementation of Self-adaptive Leaky Integrate-and-Fire Neuron. *41st Chinese Control Conference (CCC)*, Hefei, China, pp. 6989-6993. https://doi.org/10.23919/CCC55666.2022.9902118

Fang, H., Shrestha, A., Zhao, Z., Li, Y., & Qiu, Q. (2019, November). An event-driven neuromorphic system with biologically plausible temporal dynamics. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)* (pp. 1-8). IEEE. https://doi.org/10.1109/ICCAD45719.2019.8942083

Fidjeland, A. K., & Shanahan, M. P. (2010, July). Accelerated simulation of spiking neural networks using GPUs. In *The 2010 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.

https://doi.org/10.1109/IJCNN.2010.5596678

Furber, S., & Bogdan, P. (Eds.) (2020). *SpiNNaker: A spiking neural network architecture*. Now Publishers Inc. https://library.oapen.org/handle/20.500.12657/47874

Gerstein, G. L., & Mandelbrot, B. (1964). Random walk models for the spike activity of a single neuron. *Biophysical journal*, *4*(1), 41-68. https://doi.org/10.1016/s0006-3495(64)86768-0

Gerstner, W. (2008). Spike-response model. *Scholarpedia*, *3*(12), 1343. http://dx.doi.org/10.4249/scholarpedia.1343

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

Gigante, G., Del Giudice, P., & Mattia, M. (2007). Frequency-dependent response properties of adapting spiking neurons. *Mathematical biosciences*, *207*(2), 336-351. https://doi.org/10.1016/j.mbs.2006.11.010

Häusser, M. (2000). The Hodgkin-Huxley theory of the action potential. *Nature neuroscience*, *3*(11), 1165-1165. https://doi.org/10.1038/81426

Heidarpour, M., Ahmadi, A., & Rashidzadeh, R. (2016). A CORDIC based digital hardware for adaptive exponential integrate and fire neuron. *IEEE Transactions on Circuits and Systems I: Regular Papers*, *63*(11), 1986-1996. https://doi.org/10.1109/TCSI.2016.2598161

Heidarpur, M., Ahmadi, A., & Ahmadi, M. (2020). Time Step Impact on Performance and Accuracy of Izhikevich Neuron: Software Simulation and Hardware Implementation. *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 1-5. https://doi.org/10.1109/ISCAS45731.2020.9180632

Hines, M. L., & Carnevale, N. T. (2000). Expanding NEURON's repertoire of mechanisms with NMODL. *Neural computation*, *12*(5), 995–1007. https://doi.org/10.1162/089976600300001547

Hoang, R. V., Tanna, D., Jayet Bray, L. C., Dascalu, S. M., & Harris Jr, F. C. (2013). A novel CPU/GPU simulation environment for large-scale biologically realistic neural modeling. *Frontiers in neuroinformatics*, *7*, 19. https://doi.org/10.3389/fninf.2013.00019

Izhikevich, E. M. (2004). Which model to use for cortical spiking neurons?. *IEEE transactions on neural networks*, *15*(5), 1063-1070. https://doi.org/10.1109/TNN.2004.832719

Jolivet, R., Lewis, T. J., & Gerstner, W. (2004). Generalized integrate-and-fire models of neuronal activity approximate spike trains of a detailed model to a high degree of accuracy. *Journal of neurophysiology*, *92*(2), 959-976. https://doi.org/10.1152/jn.00190.2004

Karaca, Z., Korkmaz, N., Altuncu, Y., & Kılıç, R. (2021). An extensive FPGA-based realization study about the Izhikevich neurons and their bio-inspired applications. *Nonlinear Dynamics*, *105*(4), 3529-3549. https://doi.org/10.1007/s11071-021-06647-1

Koravuna, S., Sanaullah, Jungeblut, T., Rückert, U. (2023). Digit Recognition Using Spiking Neural Networks on FPGA. In: Rojas, I., Joya, G., Catala, A. (eds) Advances in Computational Intelligence. IWANN 2023. Lecture Notes in Computer Science, vol 14134. Springer, Cham. https://doi.org/10.1007/978-3-031-43085-5_32

Kumar, J., Kumar, J., & Bhakthavatchalu, R. (2016, May). Design and implementation of Hodgkin and Huxley spiking neuron model on FPGA. In *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)* (pp. 1483-1487). IEEE. https://doi.org/10.1109/RTEICT.2016.7808078

Maass, W. (1997). Networks of spiking neurons: the third generation of neural network models. Neural networks, 10(9), 1659-1671. https://doi.org/10.1016/S0893-6080(97)00011-7

McKennoch, S., Voegtlin, T., & Bushnell, L. (2009). Spike-timing error backpropagation in theta neuron networks. *Neural computation*, *21*(1), 9-45. https://doi.org/10.1162/neco.2009.09-07-610

Mutch, J., Knoblich, U., & Poggio, T. (2010). CNS: a GPU-based framework for simulating cortically organized networks. *Massachusetts Institute of Technology, Cambridge, MA, Tech. Rep. MIT-CSAIL-TR-2010-013/CBCL-286*.

Neil, D., & Liu, S. (2014). Minitaur, an Event-Driven FPGA-Based Spiking Network Accelerator. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *22*, 2621-2628. https://doi.org/10.1109/TVLSI.2013.2294916

Niu, M.C., Nandyala, S.K., Sohn, W.J., & Sanger, T.D. (December 2012). Multi-scale Hyper-time Hardware Emulation of Human Motor Nervous System Based on Spiking Neurons using FPGA. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (PP. 37-45). NIPS'12.

Niedermeier, L., Chen, K., Xing, J., Das, A., Kopsick, J., Scott, E., Sutton, N., Weber, K., Dutt, N. and Krichmar, J.L., (July 2022). CARLsim 6: an open-source library for large-scale, biologically detailed spiking neural network simulation. In *2022 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-10). IEEE. https://doi.org/10.1109/IJCNN55064.2022.9892644

Rosado-Muñoz, A., Bataller-Mompeán, M., & Guerrero-Martínez, J. (2012). FPGA implementation of spiking neural networks. *IFAC Proceedings Volumes*, *45*(4), 139-144. https://doi.org/10.3182/20120403-3-DE-3010.00074

Sanaullah, Koravuna, S., Rückert, U., Jungeblut, T. (2022). SNNs Model Analyzing and Visualizing Experimentation Using RAVSim. In: Iliadis, L., Jayne, C., Tefas, A., Pimenidis, E. (eds) Engineering Applications of Neural Networks. EANN 2022. Communications in Computer and Information Science, vol 1600. Springer, Cham. https://doi.org/10.1007/978-3-031-08223-8_4

Schwiening, C. J. (2012). A brief historical perspective: Hodgkin and Huxley. *The Journal of physiology*, *590*(Pt 11), 2571. https://doi.org/10.1113%2Fjphysiol.2012.230458

Shama, F., Haghiri, S., & Imani, M. A. (2020). FPGA realization of Hodgkin-Huxley neuronal model. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *28*(5), 1059-1068. https://doi.org/10.1109/TNSRE.2020.2980475

Stimberg, M., Brette, R., & Goodman, D. F. (2019). Brian 2, an intuitive and efficient neural simulator. *elife*, *8*, e47314. https://doi.org/10.7554/eLife.47314

Tuckwell, H. C. (1988). Introduction to Theoretical Neurobiology. Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9780511623202

Wang, Q., Li, Y., Shao, B., Dey, S., & Li, P. (2017). Energy efficient parallel neuromorphic architectures with approximate arithmetic on FPGA. *Neurocomputing*, *221*, 146-158. https://doi.org/10.1016/j.neucom.2016.09.071

Wang, Y., Taylan, O., Alkabaa, A.S., Ahmad, I., Tag-Eldin, E., Nazemi, E., Balubaid, M., & Alqabbaa H.S. (2022). An Optimization on the Neuronal Networks Based on the ADEX Biological Model in Terms of LUT-State Behaviors: Digital Design and Realization on FPGA Platforms. *Biology*, *11*, 1125. https://doi.org/10.3390/biology11081125

Yaghini Bonabi, S., Asgharian, H., Safari, S., & Nili Ahmadabadi, M. (2014). FPGA implementation of a biological neural network based on the Hodgkin-Huxley neuron model. *Frontiers in neuroscience*, *8*, 379. https://doi.org/10.3389/fnins.2014.00379

Yang, S., Liu, P., Xue, J., Sun, R., & Ying, R. (2020, October). An efficient fpga implementation of Izhikevich neuron model. In *2020 International SoC Design Conference (ISOCC)* (pp. 141-142). IEEE. https://doi.org/10.1109/ISOCC50952.2020.9333014