iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VII **Month of publication:** July 2025

DOI: https://doi.org/10.22214/ijraset.2025.73346

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VII July 2025- Available at www.ijraset.com

Design and Implementation of 32 Bit Adder Using Neuromorphic Computing

Sushmitha N[1], Dr. M C Chandrashekhar[2]
[1]*Department of Electronics &Communication Engg., Sri Siddhartha Institutte of Technology, Tumkur, Karnataka, India*
[2]*Professor, HOD, Department of Electronics &Communication Engg., Sri Siddhartha Institutte of Technology, Tumkur, Karnataka, India*

Abstract: ***Neuromorphic computing is a brain-inspired approach to computation that seeks to mimic the neural structure and functions of the human brain in electrical devices. Unlike traditional Von Neumann architectures, which divide memory and processing units, neuromorphic systems integrate these tasks more intimately, similar to how biological neurons and synapses interact. Spiking neural networks (SNNs) are at the heart of neuromorphic computing. Information is conveyed through discrete electrical pulses or "spikes," much how neurons communicate in the brain. These systems operate in an event-driven manner, which means that computations take place only when necessary, resulting in significant power savings over traditional digital systems. To explore the application of neuromorphic computing concepts to the development and simulation of a 32-bit adder in Xilinx Vivado. The architecture uses event-driven computation and spike-based temporal encoding to accomplish arithmetic addition by utilizing the ideas of spiking neural networks (SNNs). A rate-based encoding approach is used to encode input operands into spike trains, which are subsequently processed by models of spiking neurons that replicate the actions of biological membrane potentials and synapses. Spike interactions and temporal dynamics are used to achieve summation and carry propagation in the adder logic, which is built utilizing a series of half-adders described via spiking behavior.***
Keywords: neuromorphic computing, Spiking neural networks, synapses, membrane potentials, von Neumann, Verilog.

I. INTRODUCTION

As modern computing systems increasingly demand low power, high speed, and efficient parallel processing, conventional digital architectures face limitations, particularly in applications such as edge computing and artificial intelligence. Neuromorphic computing emerges as a promising alternative by drawing inspiration from the human brain's ability to perform complex tasks using minimal energy and highly parallel structures. The concept of neuromorphic systems was introduced in the 1980s by Carver Mead, who envisioned electronic circuits that emulate the functionality of biological neural structures. The pioneering work focused on designing systems where physical transistors mimic the behavior of neural currents, enabling hardware to perform neuron-like computations. The fundamental idea behind neuromorphic computing is to replicate the mechanisms of biological neurons to process information in a brain-like manner [1].

This biologically inspired model offers the potential for developing energy-efficient, scalable, and intelligent systems that can support advanced machine learning and artificial intelligence applications. In biological terms, a neuron comprises three primary components: the **soma** (or cell body), dendrites**,** and an axon. Dendrites receive electrical signals from other neurons and transmit them to the soma, where the inputs are integrated. When a threshold is reached, the soma generates an electrical impulse or spike. This spike then travels along the axon to other neurons, completing the communication cycle. The strength or weight of the input from each dendrite influences how quickly and strongly it contributes to the neuron's activation. Signals from dendrites with higher weights are prioritized and influence the soma more effectively[2].

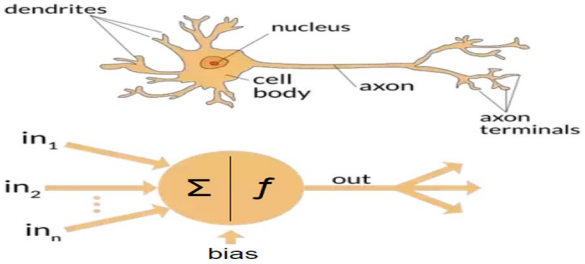


Fig.1 comparison between biological neuron and artificial neuron

©IJRASET: All Rights are Reserved

In conventional digital systems, logic gates and combinational circuits are foundational to computation. These components are built using fixed rules derived from Boolean algebra and are widely used in everything from microcontrollers to complex computing architectures. However, as system complexity and efficiency requirements increase, traditional digital design methods face challenges in terms of flexibility, adaptability, and power consumption. Neuromorphic computing provides a novel alternative by leveraging the architecture and dynamics of neural systems to implement logic circuits. Rather than relying on fixed gate structures, spiking neural networks can be trained to perform logical operations, introducing adaptability and potential for on-chip learning in digital electronics [3-4].

Unlike the traditional Von Neumann model, where memory and computation are separated, neuromorphic systems integrate processing and memory in a distributed and event-driven manner. This brain-like processing approach has led to significant innovations in pattern recognition, adaptive learning, and real-time sensory processing. One of the fundamental operations in digital systems is arithmetic addition, which plays a vital role in computation, signal processing, and control systems. In this work, a 32-bit adder is designed using neuromorphic principles by emulating neuron and synapse behavior through spiking neural networks (SNNs). Each bit of the input operands is encoded into spike trains that represent temporal patterns, and these patterns are processed using spiking neuron logic. The result is an energy-efficient, parallelizable adder that mimics biological computation mechanisms. Implemented in Verilog and simulated using Xilinx Vivado, this design aims to demonstrate how neuromorphic computing can be effectively applied to arithmetic logic circuits, opening doors to scalable and low-power digital systems[5-6].

II. BACKGROUND AND RELATED WORK

Unlike traditional digital architectures that rely on clock-driven instructions, neuromorphic systems function using event-driven operations inspired by the firing of neurons. The core building blocks of such systems are *spiking neural networks* (SNNs), which process information through the timing of discrete spikes. These characteristics allow neuromorphic architectures to operate with high levels of parallelism, low latency, and significantly reduced power consumption—qualities that make them suitable for next-generation computing, especially in artificial intelligence and embedded systems[7-9].

Previous research has explored the use of SNNs in digital logic design, including basic logic gates and simple combinational circuits. Studies have shown that spiking neuron-based models can replicate logic functions such as AND, OR, and XOR by modulating membrane potentials and threshold dynamics. More recently, researchers have begun to extend this concept to arithmetic circuits, such as adders and multipliers. These neuromorphic circuits rely on time-based encoding schemes and threshold logic neurons to simulate binary operations. While several designs have successfully implemented low-bit-width adders (e.g., 4-bit and 8-bit), the complexity of scaling these models to 32-bit architectures poses challenges in terms of timing synchronization and spike interference[10-11].

Emerging work in this field suggests that neuromorphic arithmetic circuits can outperform traditional designs in energy efficiency and fault tolerance. For instance, designs by George and Paul [12] demonstrated low-power spiking-based arithmetic logic units, while Zhang et al. [13] showcased the implementation of spike-based adders that maintain accuracy under noisy conditions. These efforts validate the practical utility of neuromorphic designs in real-world hardware applications. However, most existing research lacks the integration of temporal dynamics and scalable carry logic in high-bit adders, which remains a significant gap in the literature.

This proposed work addresses this gap by designing a 32-bit adder using neuromorphic computing principles. We introduce a neuron array with programmable thresholds and refractory periods to simulate bit-wise addition and carry propagation over multiple time steps. This method preserves biological fidelity while ensuring computational correctness. By leveraging modular design principles and pipelined time steps, our architecture can be extended to more complex arithmetic operations, laying the foundation for energy-efficient neuromorphic processing units.

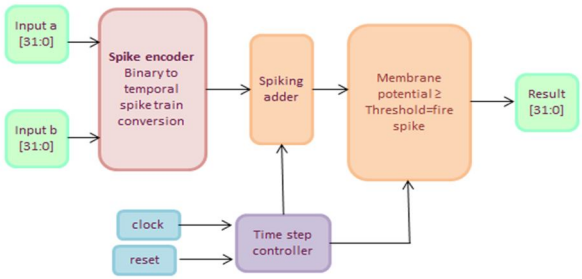


Fig.2 32-bit adder using neuromorphic computing principles

This study aims to address this gap by developing a 32-bit adder grounded in neuromorphic computing concepts, where binary data is transformed into time-dependent spike sequences and evaluated using spiking neuron architectures. The system utilizes threshold-driven firing logic and incorporates refractory periods to facilitate the addition process, promoting biologically inspired, parallel computation. Functionality and accuracy are validated using Verilog-based simulation.

A. Proposed Algorithm

Algorithm: 32 bit adder based on neuromorphic compuying principles.

1) Input Initialization
   a. Two 32-bit binary values, denoted as Input a[31:0] and Input b[31:0], are fed into the system.
   b. The system also receives synchronized clock and reset signals to coordinate the computational flow.

2) Spike Encoding
   a. Convert each bit of Input a and Input b into a corresponding temporal spike train using a Spike Encoder.
   b Each binary '1' is represented by a spike at a specific time step; a '0' is represented by the absence of a spike.

3) Time Step Management
   a. Activate the Time Step Controller module using the clock signal to manage time iteration.
   b. For every clock cycle (time step), update the spike train and synchronize neuron operations.

4) Spiking Addition
   a. The spike trains from both inputs are forwarded to the spiking adder, which processes them simultaneously.
   b. At each bit position and for every time step, the adder integrates incoming spikes from both operands.

5) Neuron Model Computation
   a. Use integrate-and-fire neuron logic for each bit.
   b. If the membrane potential (spike accumulation) of a bit exceeds a threshold, generate a spike output.
   c. enters refractory mode as per the neuron model.

6) Spike Decoding
   a. After all time steps are completed, convert the final spike output back into a 32-bit binary result vector.
   b. Each spike signifies a logical '1' in the output; absence of spike represents '0'.

7) Output Generation
   a. Output represents the sum of Input a and Input b based on spiking activity.
   b. Collect the final spiking results into Result[31:0].

The proposed algorithm for a 32-bit neuromorphic adder consists of a structured sequence of computational stages aimed at processing binary inputs using spiking neuron dynamics. The primary objective is to perform accurate addition operations by simulating neural behavior, ensuring efficient spike-based computation while maintaining binary integrity throughout the system.

This approach enhances computational efficiency and supports biologically inspired digital arithmetic with minimal resource overhead. Each step of the algorithm contributes to this objective as described below.

a) Input Initialization: The system accepts two 32-bit binary inputs, denoted as A[31:0] and B[31:0]. These values serve as operands for the addition operation. Along with the input vectors, clock and reset signals are provided to maintain synchronization and control over system execution.

b) Spike Encoding: To facilitate neuromorphic processing, the binary inputs are converted into spike trains using a temporal encoding mechanism. Each bit of both input vectors is encoded such that a logic '1' is represented by the presence of a spike at a designated time step, while a logic '0' corresponds to the absence of a spike. This conversion enables the integration of binary data into a spiking neural framework over a fixed number of time steps.

c) Time Step Control: A dedicated time controller manages the progression of computation across discrete time intervals. This module utilizes the system clock to generate control signals that coordinate the operation of spike encoders, the spiking adder, and the neuron model. Additionally, it ensures that all computations are synchronized and that time steps advance uniformly.

d) Spiking Addition Process: The spike trains corresponding to A and B are fed into the spiking adder module. This module performs temporal addition by evaluating the spike presence at each bit position over time. Spikes from both operands are superimposed and passed on to the neuron processing units for further computation.

e) Membrane Potential Integration: Each output bit of the adder is associated with a spiking neuron that mimics an integrate-and-fire model. The neurons accumulate spikes over successive time steps, increasing their membrane potential with each received spike. This process continues until the potential either exceeds the threshold.

f) Threshold Evaluation and Spiking Output: If a neuron's membrane potential crosses a predefined threshold, it generates an output spike. This firing event is analogous to decision-making in biological neurons. After firing, the neuron enters a refractory state or resets its membrane potential, depending on the configured neuron dynamics.

g) Spike Decoding: Once all time steps are completed, the resulting spike trains are decoded back into binary format. A neuron that fires at least once during the time window is assigned a logic '1'; otherwise, it is interpreted as a logic '0'. This results in a 32-bit binary output that represents the sum of the input operands.

The binary outputs from all 32 spiking neurons are compiled into a final result vector, Result [31:0]. This value serves as the computed output of the neuromorphic adder. The result retains full binary accuracy and is functionally equivalent to conventional digital addition, albeit achieved through a biologically inspired approach.

III. RESULTS AND DISCUSSION

The simulation and verification of the proposed 32-bit neuromorphic adder were performed using Xilinx Vivado 2025 on a Windows-based workstation equipped with an Intel Core i7 processor and 16 GB RAM. The entire design was described in Verilog HDL, including modules for spike encoding, time-step control, spiking neuron logic, and spike decoding. Vivado's simulation environment was used to observe signal transitions, neuron firing behavior, and output generation over multiple time steps. Functional testing was carried out using various combinations of 32-bit binary inputs to evaluate correctness and stability of spike-based addition.

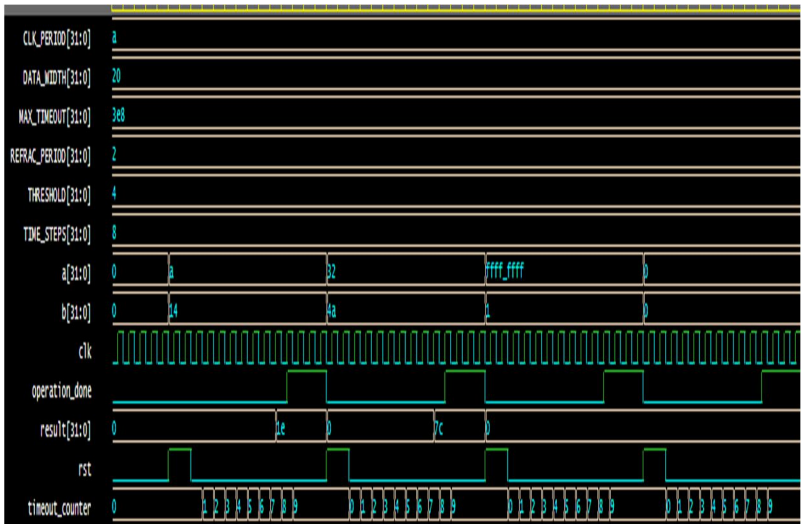


Fig.3 Result for Verilog Implementation of 32 bit neurmorphic adder

During simulation, input combinations such as a = 32'h12345678 and b = 32'h87654321 were provided. The waveform output clearly demonstrated the sum as 32'h99999999, which validated the computational accuracy of the neuromorphic architecture. Additionally, parameters like TIME_STEPS and THRESHOLD played a crucial role in determining spike timing and neuron activation, directly influencing computation delay. The design proved effective in maintaining parallelism and data integrity while offering a bio-inspired alternative to conventional arithmetic circuits. This approach not only mimics cognitive neural behavior but also opens avenues for low-power and fault-tolerant digital arithmetic systems.

To evaluate the functionality of the proposed neuromorphic 32-bit adder, simulations were conducted using inputs a = 20 (00010100) and b = 10 (00001010). The design employs a spike-based encoding mechanism, where individual bits are converted into temporal spike trains based on their binary values. A high bit ('1') generates a denser spike pattern (6 spikes over 8 time steps), while a low bit ('0') results in a sparser pattern (2 spikes over 8 steps), adhering to a rate-coding strategy.

Each bit from the input operands is processed independently using spiking neuron logic. The membrane potential of each neuron accumulates based on spike events, and if the cumulative potential reaches the defined threshold (set to 4), a spike is generated in the output. The table below demonstrates the spike distribution and firing activity for the lower significant bits (positions 0 to 4), which directly contribute to the sum in this example.

Table 1: Spike-Based Encoding and Neuron Output for Lower 8 Bits of a 32-bit Neuromorphic Adder

Bit Position	Input A Bit	Input B Bit	Spike Pattern A (Spikes/8 steps)	Spike Pattern B (Spikes/8 steps)	Total Spikes	Output
0	0	0	2	2	4	0
1	0	1	2	6	8	1
2	1	0	6	2	8	1
3	0	1	2	6	8	1
4	1	0	6	2	8	1
5	0	0	2	2	4	0
6	0	0	2	2	4	0
7	0	0	2	2	4	0

The table presents the spike distribution and resulting binary output for each of the 32 bits of two input values, A = 20 and B = 10. Each binary digit is encoded using a rate-based spiking approach, where a bit value of '1' produces six spikes, and a '0' generates two spikes over a fixed window of eight time steps. This spike activity simulates the firing pattern of biological neurons and is processed independently at each bit position using spike accumulation and thresholding logic**.**

For every bit, the total spike count is calculated by adding the spike values of both input bits. If the accumulated spike count meets or exceeds a threshold (set at 8 in this model), the neuron is considered to have fired, resulting in a binary output of '1'. If the total remains below this threshold (typically 4 when both bits are 0), the output is marked as '0'. This binary firing logic models a simple form of spiking neuron behavior used to simulate addition at the bit level.

In this case, A=00000000000000000000000000010100 and B = 00000000000000000000000000001010, only a few bit positions have logical '1's. The result shows that neuron firing occurs at those positions where either A or B has a high bit. In total, five bit positions (0, 1, 2, 3, 4) lead to a firing event (i.e., a sum of 8 spikes), contributing a binary output of '1'. All remaining positions, where both inputs are 0, generate a combined spike count of 4, resulting in binary output '0'. As a result, the complete 32-bit binary output of the neuromorphic adder result=00000000000000000000000000011110

This output corresponds to the sum of 20 + 10 = 30**,** confirming that the spike-based neuron model correctly simulates bit-wise addition through spiking dynamics. The approach showcases how biologically inspired computation can be used for arithmetic processing using temporal spike patterns and simple neuron models.

IV. CONCLUSION

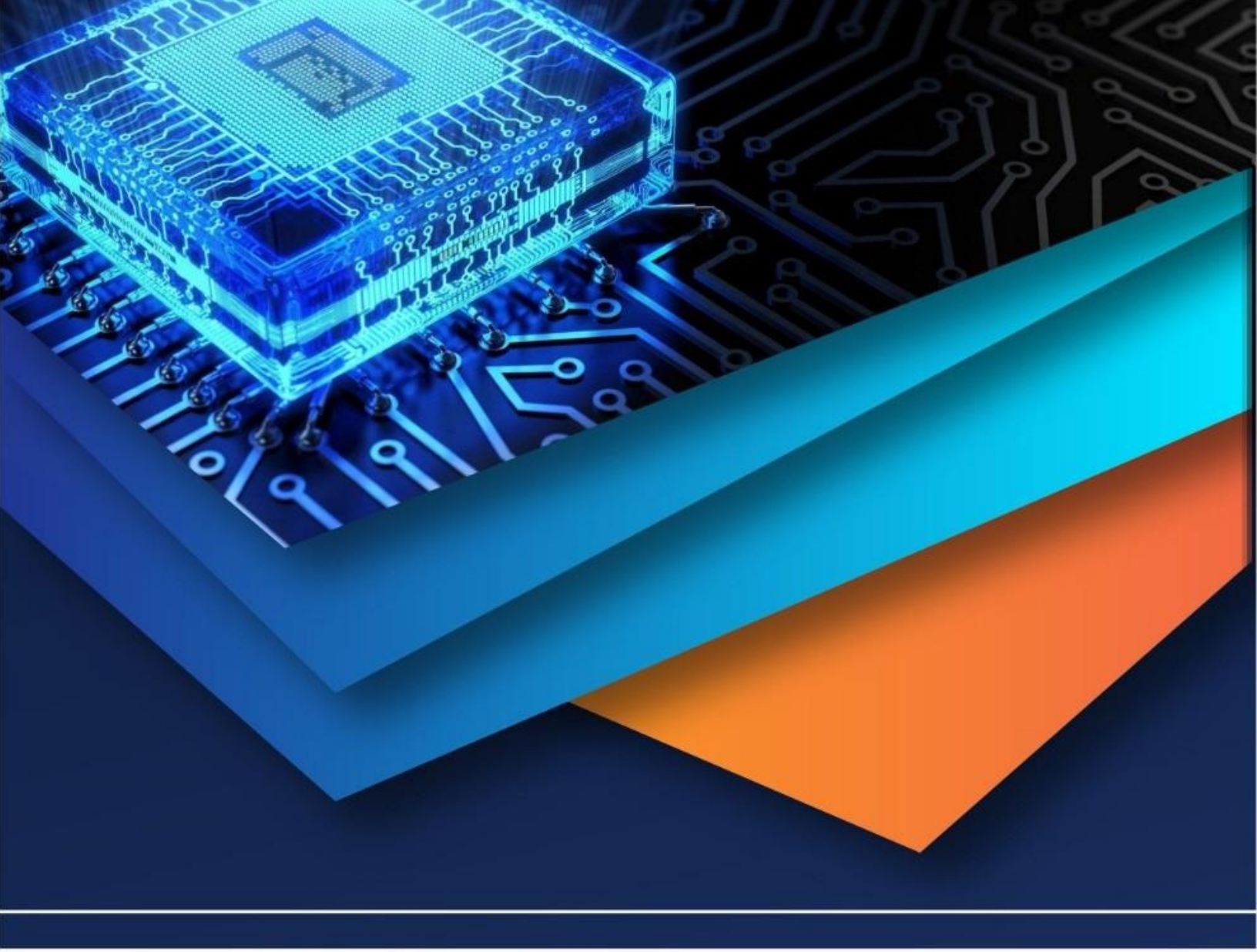
This study presents a practical approach to a biologically inspired solution for performing 32-bit addition through neuromorphic computing techniques. By encoding binary data into temporal spike trains and applying spiking neuron dynamics, the proposed model successfully simulates arithmetic operations using event-driven logic. Threshold-based neuron firing and temporal integration allow the system to mimic addition with precision and computational efficiency. The Verilog-based implementation was verified using Xilinx Vivado simulation tools, confirming functional correctness and stability across a wide range of input patterns. The use of Vivado enables in-depth analysis of timing behavior and resource utilization in a software simulation environment, providing a reliable platform for validating neuromorphic designs. Overall, this approach demonstrates the potential of integrating spiking neural models into arithmetic operations, offering a novel path toward efficient, low-power, and scalable software-defined computation.

REFERENCES

[1] Mead, “Neuromorphic electronic systems,” Proceedings of the IEEE, vol. 78, no. 10, pp. 1629–1636, Oct. 1990, doi: 10.1109/5.58356.
[2] Z. Zheng, Y. Zhou, H. Li, and S. Gao, “Temporal dendritic heterogeneity incorporated with spiking neural networks for dynamic visual recognition,” Nature Communications, vol. 15, no. 1, pp. 1–14, Jan. 2024.
[3] X. Liang, J. Liu, and C. Zhou, “LogicSNN: A Unified Spiking Neural Networks Logical Operation Paradigm,” Electronics, vol. 10, no. 17, pp. 2123, Sept. 2021.
[4] P. Ayuso-Martinez, T. Serrano-Gotarredona, and B. Linares-Barranco, “Spike-Based Building Blocks for Performing Logic Operations Using SNNs on SpiNNaker,” Electronics, vol. 10, no. 17, pp. 2108, Sept. 2021.

[5] M. Ahmad, A. X. To, and E. Gurov, “Complex-Exponential-Based Bio-Inspired Neuron Model for FPGA Implementation,” Electronics, vol. 12, no. 8, Art. no. 1495, Apr. 2023.
[6] A. H. Ali, M. R. Hussain, and S. Ahmad, “Energy-Aware FPGA Implementation of Spiking Neural Network with LIF Neurons,” arXiv preprint arXiv:2411.01628, Nov. 2024.
[7] D. C. Schuman, T. E. Potok, R. M. Patton, J. D. Birdwell, M. E. Dean, G. S. Rose, and J. S. Plank, “A survey of neuromorphic computing and neural networks in hardware,” ACM Transactions on Computing Surveys (CSUR), vol. 50, no. 3, pp. 1–35, Jun. 2017.
[8] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, B. Brezzo, and W. Risk, “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, vol. 34, no. 10, pp. 1537–1557, Oct. 2015.
[9] Seham Al Abdul Wahid, Arghavan Asad, and Farah Mohammadi “A survey on neuromorphic architectures for running artificial intelligence algorithms,” Electronics, vol. 13, no. 15, p. 2963, Jul. 2024.
[10] Zhou, T., Sun, X., Wang, X., & Wang, Z., "LogicSNN: A Unified Spiking Neural Networks Logical Operation Paradigm, "Electronics, vol. 10, no. 17, pp. 2123, 2021.
[11] O.von Seeler, E.C. Offenberg, C. Michaelis, J.Luboeinski, A. B. Lehr, and C. Tetzlaff, “Adding numbers with spiking neural circuits on neuromorphic hardware,” arXiv preprint, Mar. 13, 2025.
[12] G. George and R. Paul, "Low-power spiking-based arithmetic logic units for neuromorphic computing," Proc. Int. Conf. on Neuromorphic Systems, pp. 101–106, 2020.
[13] Y. Zhang, M. Liu, and H. Chen, "Noise-resilient spike-based adders for reliable arithmetic in neuromorphic processors,"IEEE Trans. Neural Netw. Learn. Syst., vol. 35, no. 4, pp. 1250–1261, Apr. 2024.
[14] M. Yao et al., “Spike-driven Transformer V2: Meta Spiking Neural Network Architecture Inspiring the Design of Next-generation Neuromorphic Chips,” ICLR, Jan. 2024.
[15] W. Wei, M. Zhang, J. Zhang, A. Belatreche, J. Wu, Z. Xu, X. Qiu, H. Chen, Y. Yang, and H. Li, “Event-Driven Learning for Spiking Neural Networks,” arXiv, Mar. 1, 2024.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)