FBU-DAE 2022
2 (2) : 152-161

FPGA BASED RECONFIGURABLE IMPLEMENTATIONS OF SPIKING NEURAL NETWORKS: A SHORT REVIEW

İĞNECİKLİ SİNİR AĞLARININ FPGA TABANLI YENİDEN YAPILANDIRILABİLİR UYGULAMALARI: MİNİ DERLEME

Oğuzhan YILDIRIM*, Özden NİYAZ**, Burcu ERKMEN***

ABSTRACT

Due to the increasing data capacity, low power consumption, and high-speed data processing expectations of systems in our daily lives today, the Von Neumann bottleneck has become a more important problem than in the past. For these reasons, conventional computer architectures can no longer fully meet today's requirements. Neuromorphic designs have been considered an alternative solution for all, as they can mimic the human brain in terms of processing large amounts of data quickly with low power consumption. Although the success of traditional Artificial Neural Network methods is satisfactory, biological systems are still much more advantageous in terms of power consumption. Neuromorphic hardware architectures based on Spiking Neural Network (SNN), which are the most biologically plausible and are referred to as third-generation neural networks, overcome the Von Neumann bottleneck and provide a more suitable hardware structure for intelligent systems. The use of reconfigurable hardware for the implementation of neuromorphic architectures creates a faster and more updatable research field than integrated circuits and computational approaches. Therefore, this study has reviewed FPGA-based reconfigurable implementations of Spiking Neural Networks (SNN) in the literature and compared these studies in terms of power consumption, learning capability, resource consumption, and accuracy**.**

Keywords: Spiking Neural Networks, Reconfigurable Implementations, FPGA, Neuromorphic

Geliş Tarihi/Received: 23 Nisan 2022
Kabul Tarihi/Accepted: 16 Mayıs 2022

Derleme Makalesi/Review Article

*
Elektronik ve Haberleşme Mühendisliği, Yıldz Teknik Üniversitesi, İstanbul/Türkiye

Yıldız Technical University, Istanbul / Turkey

ORCID: 0000-0003-3898-5610
**
Elektronik ve Haberleşme Mühendisliği, Yıldz Teknik Üniversitesi, İstanbul/Türkiye

Yıldız Technical University, Istanbul / Turkey

ORCID: 0000-0001-5550-2081
***
Elektronik ve Haberleşme Mühendisliği, Yıldz Teknik Üniversitesi, İstanbul/Türkiye

Yıldız Technical University, Istanbul / Turkey

ORCID: 0000-0002-5581-9764

ÖZET

Günümüzde, günlük hayatımızda sistemlerin artan veri kapasitesi, düşük güç tüketimi ve yüksek hızlı veri işleme beklentileri nedeniyle Von Neumann darboğazı geçmişe göre daha önemli bir sorun haline gelmiştir. Bu nedenle, geleneksel bilgisayar mimarileri artık günümüzün gereksinimlerini tam olarak karşılayamamaktadır. Nöromorfik tasarımlar, düşük güç tüketimi ile büyük miktarda veriyi hızlı bir şekilde işleme açısından insan beynini taklit edebildiklerinden, alternatif bir çözüm olarak görülmektedir. Geleneksel Yapay Sinir Ağı yöntemlerinin başarısı tatmin edici olsa da, biyolojik sistemler güç tüketimi açısından hala çok daha avantajlıdır. Biyolojik olarak en gerçekçi ve üçüncü nesil sinir ağları olarak anılan İğnecikli Sinir Ağlarına dayalı nöromorfik donanım mimarileri, Von Neumann darboğazını aşarak akıllı sistemler için daha uygun bir donanım yapısı sağlamaktadır. Nöromorfik mimarilerin uygulanması için yeniden yapılandırılabilir donanımın kullanılması, entegre devreler ve hesaplama yaklaşımlarından daha hızlı ve güncellenebilir bir araştırma alanı yaratmaktadır. Bu sebeple, bu çalışma kapsamında, literatürdeki İğnecikli Sinir Ağlarının FPGA tabanlı yeniden yapılandırılabilir uygulamaları gözden geçirilmiş ve bu çalışmalar güç tüketimi, öğrenme yeteneği, kaynak tüketimi ve doğruluk oranları açısından karşılaştırılmıştır.

Anahtar Kelimeler: İğnecikli Sinir Ağları, Yeniden Yapılandırılabilir Uygulamalar, FPGA, Nöromorfik

1. INTRODUCTION

The foundations of computers are usually based on the traditional Von Neumann architecture, where the processing unit and the memory units are separated (Neumann,1993). Hardware with this structure works by taking the data from the memory unit before the calculation of the processing unit and then sending it back to this unit. Artificial intelligence applications, which contain very complex operations and use large datasets, are faced with the Von Neumann bottleneck today. The fact that the data is received from a separate memory unit over a limited speed bus and reaches the processing unit and the results are sent back to the memory unit, in the same way, limits the artificial intelligence applications performed on the hardware in terms of energy consumption and data transfer rate (Xiong et. al., 2020). As intelligent systems using

artificial neural networks are becoming more common in our lives, new methods that can overcome this bottleneck are needed to be examined. (Sun et. al., 2021; Schuman et. al., 2017). Digital or analog hardware that implements neural systems is commonly referred to as neuromorphic structures. To solve the bottleneck issue, neuromorphic structures are used as the hardware to provide solutions for face recognition, object classification, speech recognition, and natural language processing (Lemaire et. al., 2020; Miró-Amarante et. al., 2017; Pasupathi et. al., 2014). Spiking Neural Networks (SNN), referred to as third-generation ANN, are commonly used in neuromorphic designs (Maas,1997). Due to their biological plausibility, SNN algorithms have the advantages of processing speed and low power consumption similar to the human brain. As the novel intelligent system architectures are required to keep memory and processing units together, implementing neuromorphic structures will help them to be successfully realized by silicon-based solutions on hardware (Yan et. al., 2019). To reduce the compelling impact of silicon-based designs in the process, some studies have been conducted on FPGA. Moreover, various problems such as feature extraction, autonomous vehicles, and letter recognition are solved on FPGA using SNN (Sun et. al., 2017; Zhao et. al.,2022; Humaidi et. al., 2020). Another reason for using FPGA-based SNN studies can be seen in Table1, where it is clearly shown that SNN achieves higher classification results than traditional ANN methods on the same dataset (Si et. al., 2022; Fang et. al., 2020). In this study, FPGA-based implementations of SNN have been perused and examined in terms of power consumption, learning ability, resource consumption, and accuracy. The overall flow of this paper is organized as follows; Section 2 mainly focuses on explaining the SNN used in the related works, while Section 3 demonstrates the reconfigurable implementations of SNN. Section 4 presents the application areas of the reconfigurable studies, and finally, Section 5 summarizes the conclusions about FPGA-based implementations of SNN.

Table 1. Comparison of FPGA Based Implementations of MLP and SNN

Neural Network	FPGA Platform	Precision	Num. of Training Data	Num. of Test Data	Max. Accuracy rate	Resource Utilization
MLP (Si et. al., 2022)	Cyclone IVE	8-bit fixed	55000	10000	89%	34k Logic Units Overall: 29%
SNN (Fang et. al.,2020)	Zynq Ultrascale+ XCZU9EG	16-bit fixed	60000	10000	99.42%	LUT: 56.9% BRAM: 30.9% DSP : 71.2%
SNN (Wang et.al., 2017)	Virtex-6 FPGA	16-bit fixed	60000	10000	89,1%	LUTs: 97,287 FFs: 58,826 BRAM : 34

2. SPIKING NEURAL NETWORKS (SNN)

In the biological nervous system, information is carried by chemical elements between neurons through synaptic connections and accumulates in the form of electrical spikes in the neurons. Briefly, the information between neurons in the brain is carried by spikes through synaptic connections. The currents which are transmitted across the synaptic connections are weighted according to learning and then accumulated in the neurons. If the accumulated membrane potential exceeds the threshold value, then a spike occurs at the postsynaptic neuron. Neurons that fire together during the learning make stronger or higher weighted connections and enable the network to produce outputs that are close to real values. Spiking Neural Networks (SNN) implements mathematical and electrical models of biological neurons and learning algorithms inspired by the human brain. The neuron models mostly used in studies of FPGA-based implementations of SNN are derived from the Leaky-Integrate Fire and Izhikevich neuron models, and the commonly used learning algorithm is Spike Time Dependent Plasticity (STDP) (Markram et. al., 2012).

2.1. Neuron Models

The Izhikevich model is a simplified neuron model, which is developed to generate spike voltages in different patterns of membrane potential. In the Izhikevich model, the

membrane potential is a dimensionless variable, represented by v in (1) and formulated with the synaptic currents supplied to the neuron and the membrane recovery variable u in (2). When the membrane potential reaches 30 mV, the neuron fires the spike and returns to its resting state, as shown in (3). a, b, c, and d are dimensionless parameters that have been used to generate different firing patterns (Izhikevich, 2003).

$$v' = 0.04v^2 + 5v + 140 - u + I \quad (1)$$

$$u' = a(bv - u) \quad (2)$$

$$if\ v \geq 30mV, then \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \quad (3)$$

In the Leaky-Integrate-Fire model, the membrane potential is generated by a simple RC circuit. The current, coming from the synapses generates the voltage u as charge and discharge across the capacitance. τ_m, represents the membrane time constant of the neuron and u_{rest} refers to the reverse voltage for leakage in (4). In the absence of cumulative input currents, $I(t)$, the membrane potential drops exponentially to the resting potential, and then spikes occur (Gerstner et. al., 2014).

$$\tau_m \frac{du}{dt} = -[u(t) - u_{rest}] + RI(t) \quad (4)$$

2.2. Learning Rule

Spike Timing Dependent Plasticity, which is based on the firing times of neurons, is the most commonly used learning algorithm for SNN (Mark et. al., 2012). The STDP learning rule increases or decreases the weights between presynaptic and postsynaptic neurons according to the difference between the firing times and strengthens the connection between neurons that fired together and vice versa. The equation (5) gives results corresponding to the exponential change in the difference of spike firing times ($t_{post} - t_{pre}$) with constants, and this function is used to determine the weight difference to be updated in (6). Consequently, the weights are updated upwards as in (7) when the postsynaptic neuron fired after the presynaptic neuron, otherwise, they are updated downwards (Iakymchuk et. al., 2015).

$$w(x) = \begin{cases} A_+ \exp(-\frac{x}{\tau_+}), & x > 0 \\ A_- \exp\left(\frac{x}{\tau_-}\right), & x < 0 \end{cases} \quad (5)$$

$$\Delta\omega_j = \sum_{k=1}^{N} \sum_{l=1}^{N} w\left(t_i^k - t_j^l\right) \quad (6)$$

$$\omega_{new} = \omega_{old} + \sigma\Delta\omega(\omega_{max} - \omega_{old}), \text{if } \Delta\omega > 0 \quad (7)$$
$$\omega_{new} = \omega_{old} + \sigma\Delta\omega(\omega_{old} - \omega_{min}), \text{if } \Delta\omega \leq 0$$

2.3. Network

According to the definition of the problem, the number of features, and the expected output value of the network; neural networks can be created through different numbers of neurons, which are formed by connecting series as various layers. Therefore, implementing SNN on FPGA provides the advantage of self-repeating structures being reconnected without the complexity and in a fault-tolerant manner. Figure 1 shows an example of a simple two-layer SNN network with two input neurons and one output neuron, where the neuron between each layer runs along a synapse that is the presynaptic neuron of the previous layer. Time-dependent spike trains, coming from the previous layer are collected on the postsynaptic neuron after being multiplied by the weight through the synapses. Once the collected spike trains exceed the membrane potential, post-synaptic neuron produce a spike.

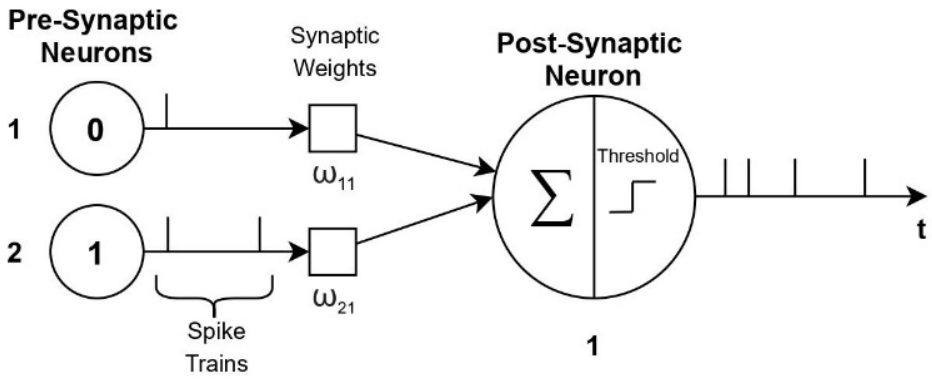

Figure 1. Simple two-layered Spiking Neural Network (SNN)

3. RECONFIGURABLE IMPLEMENTATIONS OF SNN

To overcome the complexity and time-consuming difficulties, encountered in the VLSI design processes of neuromorphic circuits, the reconfigurable FPGA structure can be considered as the most fitting solution. Reconfigurable implementations allow the algorithms to be computed quickly on the hardware and provide debugging capabilities in a shorter time. Although some of the hardware studies about SNN in literature have been implemented as ASIC (Chen et. al., 2018), FPGA plays a different role in terms of reconfigurability and energy efficiency. Thanks to the reconfigurable and parallel structure, FPGA implementations require significantly less time for both designing and testing, especially compared to ASIC design. Neuron implementations at the FPGA level are done using operation blocks, as shown in Figure 2, which implements an Izhikevich neuron with adders, multipliers, and comparators (Ambroise et. al., 2013). Reconfigurable implementations provide the ability to keep adding neurons and synapse blocks for creating a network in a much easier way, as shown in Figure 3. FPGA-based reconfigurable implementations of SNN have been created using different neuron models, learning algorithms, and network topologies (Liu et. al., 2019; Heidarpur et. al., 2019; Murali et. al., 2016; Grassia et. al., 2016).

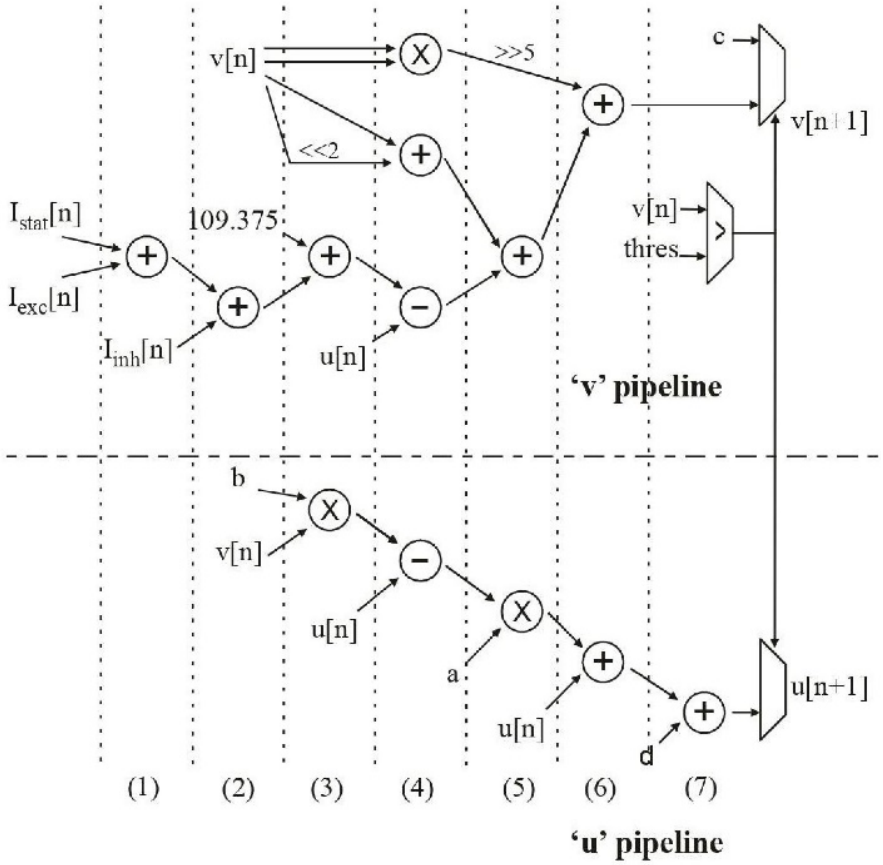

Figure 2. FPGA implementation of Izhikevich neuron (Ambroise et. al.,2013)

3.1. Network Structure

Different results can be obtained in the FPGA-based SNN studies even for the same dataset by using different neuron models and network structures. For example, in one of those studies conducted on the hardware, a pattern recognition study was performed with 1591 LIF neurons, 87.7% success was achieved on the MNIST dataset by implementing the STDP learning rule on-chip, and power consumption was detailed with respect to different levels of parallel architectures (Wang et. al., 2017). Similarly, the training has been completed on-board for the network, which has been designed by using the LIF neuron model, STDP learning algorithm and MNIST dataset and the success rate has been calculated as 85.28% on Virtex 7 (Li et. al., 2021). Examining the FPGA and software environment in terms of speed and performance rates, the network designed with the LIF neuron model and trained with STDP achieved 13.5% faster pattern recognition on the FPGA and 25.8% faster training time, compared to computational simulation of the same network (Wang et. al., 2017).

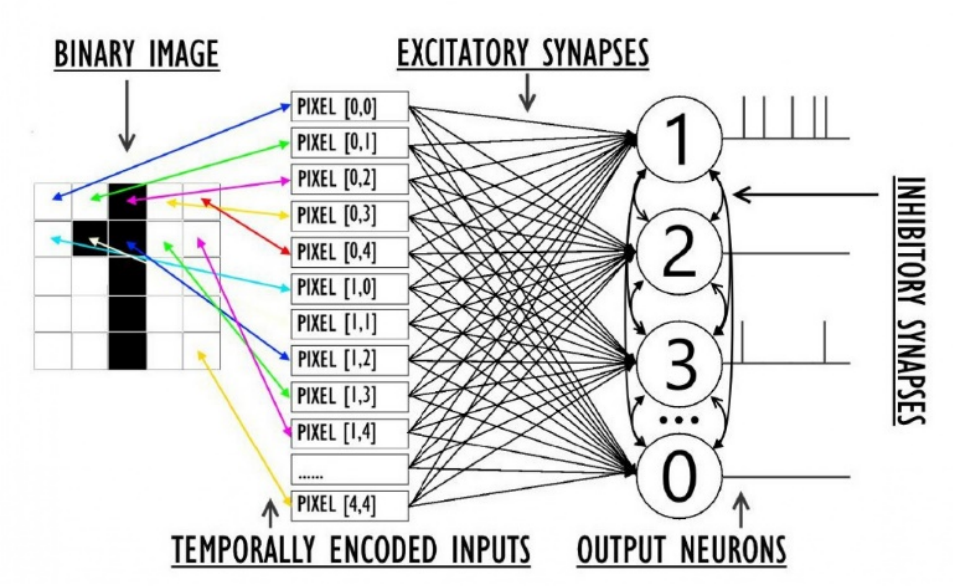

Figure 3. Reconfigurable implementation of SNN for pattern recognition (Lammie et. al.,2018)

Furthermore, SNN networks are also used for different datasets for classification. Two separate networks were created for different studies; one using the Izhikevich neuron model with application-specific 24x24 pixel images as the dataset, while the other was created using IF neurons along with edge detection, performed over custom datasets. The network structure was first simulated in the software environment and then implemented on the Xilinx XC4VFX60 (Rice et. al., 2009; Glackin et. al., 2009).

3.2. Learning Method

Two different training methods are preferred for FPGA-based implementations of SNN: Off-chip and on-chip training for hybrid or ordinary models. It has been observed that using the MNIST dataset and custom datasets, results in different performance rates for each character recognition application. From this perspective, a study using 5x5-pixel custom images as the dataset has achieved 94% accuracy with on-chip learning (Lammie et. al., 2018). The STDP learning algorithm is also used on FPGA for Spatio-temporal pattern recognition problems (Wang et. al., 2013). Moreover, with the studies in SNN, the classical ANN learning algorithms can be modified and used for off-chip training. For example, during an off-chip training study, where a feed-forward back-propagation algorithm was used for training on MNIST dataset, Izhikevich neuron model have been used to implement SNN on the Xilinx VC707 FPGA, the accuracy rate of 98% was reached and the power consumption was measured as 0.36W (Zhang et. al., 2019).

Similarly, the network created using LIF was applied to Xilinx ZC706, and the weighting was determined by threshold matching via MATLAB as off-chip learning. An accuracy rate of 97.06% was achieved and the power consumption was measured as 0.477W (Han et. al., 2020). In another study implements on-chip learning, which still uses the MNIST dataset, an accuracy rate of 90.70% was achieved with the network trained by back-propagation. Also, the power consumption was calculated as 161mW (Losh & Llamocca, 2019). Moreover, some studies try different training algorithms to achieve application-specific results. The use of the FORCE training algorithm, 510 Izhikevich Neuron shows the compatibility of FPGA with the memory-intensive structure (Sherbaf et. al., 2020). An accuracy rate of 96% was achieved in a study classified on Virtex 7 using Tempotron Supervised training algorithm with 1000 images mimicking LIF neurons and the human eye (Zhang et. al., 2020). It is possible to modify the performance rate and energy consumption factors using hybrid training models.

4. APPLICATION AREAS

FPGA implementation of SNN is studied in terms of pattern recognition and classification, which are the most common research areas for SNN.

4.1. Pattern Recognition

Considering neuromorphic reconfigurable pattern recognition studies, it is apparent that most studies were conducted with the MNIST dataset. Most of the SNN studies using FPGAs deal with the classification of MNIST dataset (LeCun et. al., 2012). The MNIST dataset consists of handwritten number images from 0 to 9. This dataset contains 60000 training and 10000 test data, which is a total of 70000 images in form of 28x28 pixels. Using different learning rules, SNN and MNIST can classify the dataset with 95% accuracy, showing that the pattern recognition can be performed on neuromorphic hardware with a high performance (Diehl & Cook, 2015). Considering the pattern recognition in terms of power consumption and accuracy; 90.70% accuracy rate and 69mW power consumption (without static power) were observed (Losh & Llamocca, 2019). While supervised learning based on a temporal coding algorithm achieved a 3.02% error rate, another SNN algorithm which was created by switching CNN network, achieved 90.2% recognition accuracy (Mostafa et. al., 2017; Tang et. al., 2020). When the SNN is created with back-propagation, the success rate increases to 99.42%, with the power consumption being measured as 4.5W (Fang et. al., 2020). In a study including the LIF neuron model and conducted on the Xilinx Spartan 6, the success rate was calculated as 92% with the power consumption as 1.5W (Neil & Liu, 2014). Reducing power consumption while increasing the pattern recognition performance rate could be considered future work.

4.2. Classification

FPGA-based SNN implementations are also used to solve classification issues other than pattern recognition with MNIST. Apart from the MNIST dataset, FPGA-based SNN implementations are also used in areas such as solving classification problems with satellite imagery (Lemaire et. al., 2020). In addition, in a study for vowel recognition, the recognition of Spanish words was completed successfully (Miró-Amarante, 2017). Unlike other studies, the spiking neurons and the spikes generated in response to the musical notes were sent to the computer through FPGA and classified on CNN which is generated on MATLAB. In the study using a Virtex 5 FPGA, accuracy rate and power consumption were measured as 97.5% and 29.7mW (Cerezuela-Escudero,2015).

5. CONCLUSION

Many factors such as the neuron model and the number of neurons used in FPGA applications; the designed network architecture, the spike coding technique, the learning algorithm, the platform, whether it is a hybrid network or not, and the dataset, affect the learning accuracy and power consumption. Therefore, while considering the studies in the literature, we aim to examine the publications with similar characteristics in terms of the dataset and the application platforms. All the applications in Table 2 use the MNIST dataset, but due to the different methods and platforms, differences in power consumption and accuracy rates were observed.

Comparing the studies in Table 2, we can see that different studies targeting the same application purpose give different results depending on the application method. As can be seen from this comparison, high accuracy rates were obtained, while the power consumption remained in terms of milliwatt level. As a result, the research on algorithms aiming to reduce the power consumption of FPGA-based SNN implementations still needs further improvement.

Since classical computers are very power-hungry due to their separate memories, neuromorphic implementations are very promising in terms of this power consumption

aspect. Thanks to the configurable structures of reconfigurable implementations, the difficulties encountered by researchers in neuromorphic hardware design are circumvented. In this review, reconfigurable implementations of SNN are investigated in terms of learning accuracy, neuron models, learning methods, and power consumption. It is possible to conclude that this rapidly evolving field of study is likely to continue presenting challenges in terms of power consumption.

Table 2. Comparison of Power Consumption and Learning Accuracy on Pattern Recognition With MNIST

Ref.	Applied Platform	Power/Energy Consumption	Learning Accuracy
(Wang et.al., 2017)	Virtex-6 FPGA	1339.83 mJ	89.1%
(Losh & Llamocca, 2019)	Zynq XC7Z010-1CLGC400C	161 mW	94.6%
(Zhang et. al., 2019)	Xilinx VC707 FPGA	0.36 W	98%
(Han et. al., 2020)	Xilinx ZC706	0.477 W	97.06%
(Neil & Liu, 2014)	Xilinx Spartan-6	1.5 W	92%

REFERENCES

Akbarzadeh-Sherbaf, K., Safari, S., & Vahabie, A.-H. (2020). A digital hardware implementation of spiking neural networks with binary FORCE training. Neurocomputing, 412, 129–142. https://doi.org/10.1016/j.neucom.2020.05.044

Ambroise, M., Levi, T., Bornat, Y., & Saïghi, S. (2013). Biorealistic spiking neural network on FPGA. 2013 47th Annual Conference on Information Sciences and Systems (CISS). https://doi.org/10.1109/ciss.2013.6616689

Cerezuela-Escudero, E., Jimenez-Fernandez, A., Paz-Vicente, R., Dominguez-Morales, M., Linares-Barranco, A., & Jimenez-Moreno, G. (2015). Musical notes classification with neuromorphic auditory system using FPGA and a convolutional spiking network. 2015 International Joint Conference on Neural Networks (IJCNN). https://doi.org/10.1109/ijcnn.2015.7280619

Chen, G. K., Kumar, R., Sumbul, H. E., Knag, P. C., & Krishnamurthy, R. K. (2018). A 4096-Neuron 1M-Synapse 3.8PJ/SOP Spiking Neural Network with On-Chip STDP Learning and Sparse Weights in 10NM FinFET CMOS. 2018 IEEE Symposium on VLSI Circuits. https://doi.org/10.1109/vlsic.2018.8502423

Diehl, P. U., & Cook, M. (2015). Unsupervised learning of digit recognition using spike-timing-dependent plasticity. Frontiers in Computational Neuroscience, 9. https://doi.org/10.3389/fncom.2015.00099

Fang, H., Mei, Z., Shrestha, A., Zhao, Z., Li, Y., & Qiu, Q. (2020). Encoding, model, and architecture. Proceedings of the 39th International Conference on Computer-Aided Design. https://doi.org/10.1145/3400302.3415608

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). Part II Generalized Integrate-and-Fire Neurons | Neuronal Dynamics online book. Retrieved April 20, 2022, from Epfl.ch website: https://neuronaldynamics.epfl.ch/online/Pt2.html

Glackin, B., Harkin, J., McGinnity, T. M., Maguire, L. P., & Wu, Q. (2009). Emulating Spiking Neural Networks for edge detection on FPGA hardware. 2009 International Conference on Field Programmable Logic and Applications. https://doi.org/10.1109/fpl.2009.5272339

Grassia, F., Kohno, T., & Levi, T. (2016). Digital hardware implementation of a stochastic two-dimensional neuron model. Journal of Physiology-Paris, 110(4), 409–416. https://doi.org/10.1016/j.jphysparis.2017.02.002

Han, J., Li, Z., Zheng, W., & Zhang, Y. (2020). Hardware implementation of spiking neural networks on FPGA. Tsinghua Science and Technology, 25(4), 479–486. https://doi.org/10.26599/tst.2019.9010019

Heidarpur, M., Ahmadi, A., Ahmadi, M., & Rahimi Azghadi, M. (2019). CORDIC-SNN: On-FPGA STDP Learning With Izhikevich Neurons. IEEE Transactions on Circuits and Systems I: Regular Papers, 66(7), 2651–2661. https://doi.org/10.1109/tcsi.2019.2899356

Humaidi, A. J., Kadhim, T. M., Hasan, S., Kasim Ibraheem, I., & Taher Azar, A. (2020). A Generic Izhikevich-Modelled FPGA-Realized Architecture: A Case Study of Printed English Letter Recognition. 2020 24th International Conference on System Theory, Control and Computing (ICSTCC). https://doi.org/10.1109/icstcc50638.2020.9259707

Iakymchuk, T., Rosado-Muñoz, A., Guerrero-Martínez, J. F., Bataller-Mompeán, M., & Francés-Víllora, J. V. (2015). Simplified spiking neural network architecture and STDP learning algorithm applied to image classification. EURASIP Journal on Image and Video Processing, 2015(1). https://doi.org/10.1186/s13640-015-0059-4

Izhikevich, E. M. (2003). Simple model of spiking neurons. IEEE Transactions on Neural Networks, 14(6), 1569–1572. https://doi.org/10.1109/tnn.2003.820440

Ju, X., Fang, B., Yan, R., Xu, X., & Tang, H. (2020). An FPGA Implementation of Deep Spiking Neural Networks for Low-Power and Fast Classification. Neural Computation, 32(1), 182–204. https://doi.org/10.1162/neco_a_01245

Lammie, C., Hamilton, T., & Azghadi, M. R. (2018). Unsupervised Character Recognition with a Simplified FPGA Neuromorphic System. 2018 IEEE International Symposium on Circuits and Systems (ISCAS). https://doi.org/10.1109/iscas.2018.8351532

LeCun, Y., Cortes, C., & Burges, C. (2012). MNIST handwritten digit database. Retrieved April 20, 2022, from Lecun.com website: http://yann.lecun.com/exdb/mnist/

Lemaire, E., Moretti, M., Daniel, L., Miramond, B., Millet, P., Feresin, F., & Bilavarn, S. (2020). An FPGA-Based Hybrid Neural Network Accelerator for Embedded Satellite Image Classification. 2020 IEEE International Symposium on Circuits and Systems (ISCAS). https://doi.org/10.1109/iscas45731.2020.9180625

Li, S., Zhang, Z., Mao, R., Xiao, J., Chang, L., & Zhou, J. (2021). A Fast and Energy-Efficient SNN Processor With Adaptive Clock/Event-Driven Computation Scheme and Online Learning. IEEE Transactions on Circuits and Systems I: Regular Papers, 68(4), 1543–1552. https://doi.org/10.1109/tcsi.2021.3052885

Liu, K., Cui, X., Zhong, Y., Kuang, Y., Wang, Y., Tang, H., & Huang, R. (2019). A Hardware Implementation of SNN-Based Spatio-Temporal Memory Model. Frontiers in Neuroscience, 13. https://doi.org/10.3389/fnins.2019.00835

Losh, M., & Llamocca, D. (2019). A Low-Power Spike-like Neural Network Design. Electronics, 8(12), 1479. https://doi.org/10.3390/electronics8121479

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

Markram, H., Gerstner, W., & Sjöström, P. J. (2012). Spike-Timing-Dependent Plasticity: A Comprehensive Overview. Frontiers in Synaptic Neuroscience, 4. https://doi.org/10.3389/fnsyn.2012.00002

Miró-Amarante, L., Gómez-Rodríguez, F., Jiménez-Fernández, A., & Jiménez-Moreno, G. (2017). A spiking neural network for real-time Spanish vowel phonemes recognition. Neurocomputing, 226, 249–261. https://doi.org/10.1016/j.neucom.2016.12.005

Mostafa, H., Pedroni, B. U., Sheik, S., & Cauwenberghs, G. (2017). Fast classification using sparsely active spiking networks. 2017 IEEE International Symposium on Circuits and Systems (ISCAS). https://doi.org/10.1109/iscas.2017.8050527

Murali, S., Kumar, J., Kumar, J., & Bhakthavatchalu, R. (2016). Design and implementation of Izhikevich spiking neuron model on FPGA. 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT). https://doi.org/10.1109/rteict.2016.7807968

Neil, D., & Liu, S.-C. (2014). Minitaur, an Event-Driven FPGA-Based Spiking Network Accelerator. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 22(12), 2621–2628. https://doi.org/10.1109/tvlsi.2013.2294916

Ou, Q.-F., Xiong, B.-S., Yu, L., Wen, J., Wang, L., & Tong, Y. (2020). In-Memory Logic Operations and Neuromorphic Computing in Non-Volatile Random Access Memory. Materials, 13(16), 3532. https://doi.org/10.3390/ma13163532

Rice, K. L., Bhuiyan, M. A., Taha, T. M., Vutsinas, C. N., & Smith, M. C. (2009). FPGA Implementation of Izhikevich Spiking Neural Networks for Character Recognition. 2009 International Conference on Reconfigurable Computing and FPGAs. https://doi.org/10.1109/reconfig.2009.77

Schuman, C. D., Potok, T. E., Patton, R. M., Douglas, B. J., Dean, M. E., Rose, G. S., & Plank, J. S. (2017). A Survey of Neuromorphic Computing and Neural Networks in Hardware. ArXiv.org. https://doi.org/10.48550/arXiv.1705.06963

Si, J., & Harris, S. L. (2018). Handwritten digit recognition system on an FPGA. 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC). https://doi.org/10.1109/ccwc.2018.8301757

Sun, B., Guo, T., Zhou, G., Ranjan, S., Jiao, Y., Wei, L., … Wu, Y. A. (2021). Synaptic devices based neuromorphic computing applications in artificial intelligence. Materials Today Physics, 18, 100393. https://doi.org/10.1016/j.mtphys.2021.100393

Sun, Q. Y., Wu, Q. X., Wang, X., & Hou, L. (2017). A spiking neural network for extraction of features in colour opponent visual pathways and FPGA implementation. Neurocomputing, 228, 119–132. https://doi.org/10.1016/j.neucom.2016.09.093

T, P., A, A. B. R., & J, A. (2014). FPGA IMPLEMENTATION OF ADAPTIVE INTEGRATED SPIKING NEURAL NETWORK FOR EFFICIENT IMAGE RECOGNITION SYSTEM. ICTACT Journal on Image and Video Processing, 4(4), 848–852. https://doi.org/10.21917/ijivp.2014.0122

Tang, H., Cho, D., Lew, D., Kim, T., & Park, J. (2020). Rank order coding based spiking convolutional neural network architecture with energy-efficient membrane voltage updates. Neurocomputing, 407, 300–312. https://doi.org/10.1016/j.neucom.2020.05.031

Von Neumann, J. (1993). First draft of a report on the EDVAC. IEEE Annals of the History of Computing, 15(4), 27–75. https://doi.org/10.1109/85.238389

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, R., Cohen, G., Stiefel, K. M., Hamilton, T. J., Tapson, J., & van Schaik, A. (2013). An FPGA Implementation of a Polychronous Spiking Neural Network with Delay Adaptation. Frontiers in Neuroscience, 7. https://doi.org/10.3389/fnins.2013.00014

Yan, Y., Kappel, D., Neumarker, F., Partzsch, J., Vogginger, B., Hoppner, S., … Mayr, C. (2019). Efficient Reward-Based Structural Plasticity on a SpiNNaker 2 Prototype. IEEE Transactions on Biomedical Circuits and Systems, 13(3), 579–591. https://doi.org/10.1109/tbcas.2019.2906401

Zhang, G., Li, B., Wu, J., Wang, R., Lan, Y., Sun, L., … Chen, Y. (2020). A low-cost and high-speed hardware implementation of spiking neural network. Neurocomputing, 382, 106–115. https://doi.org/10.1016/j.neucom.2019.11.045

Zhang, J., Wu, H., Wei, J., Wei, S., & Chen, H. (2019). An Asynchronous Reconfigurable SNN Accelerator With Event-Driven Time Step Update. 2019 IEEE Asian Solid-State Circuits Conference (A-SSCC). https://doi.org/10.1109/a-sscc47793.2019.9056903

Zhao, F., Zeng, Y., & Xu, B. (2018). A Brain-Inspired Decision-Making Spiking Neural Network and Its Application in Unmanned Aerial Vehicle. Frontiers in Neurorobotics, 12. https://doi.org/10.3389/fnbot.2018.00056