Light-weight Spiking Neuron Processing Core for Large-scale 3D-NoC based Spiking Neural Network Processing Systems

Ogbodo Mark Ikechukwu*, The H. Vu†, Khanh N. Dang‡, Abderazek Ben Abdallah§
* §*Adaptive Systems Laboratory, Graduate School of Computer Science and Engineering, The University of Aizu*
Aizu-Wakamatsu 965-8580, Fukushima, Japan
†*Faculty of Infornation Technology, Hung Yen University of Technology and Education,* Hung Yen,Vietnam.
‡*SISLAB, University of Engineering and Technology, Vietnam National University Hanoi,* Hanoi, 123106, Vietnam.
{d8211104, benab}@u-aizu.ac.jp, vuhuythe@utechy.edu.vn, khanh.n.dang@vnu.edu.vn

***Abstract*—With the increasing demand for computing machines that more closely model the biological brain, the field of neuro-inspired computing has progressed to the exploration of Spiking Neural Networks (SNN), and to best the challenges of conventional *Von Neumann* architecture, several hardware-based (neuromorphic) chips have been designed. A neuromorphic chip is based on spiking neurons which process input information only when they receive spike signals. Given a sparsely-distributed input spike train, the power consumption for such event-driven hardware would be reduced since large portions of the network that are not driven by incoming spikes can be set into a power-gated mode. The challenges that need to be solved toward building in hardware such a spiking neuromorphic chip with a massive number of synapse include building small-sized spiking neuro-cores with low-power consumption, efficient neurocoding scheme, and lightweight on-chip learning algorithm. In this paper, we present the hardware implementation and evaluation of a light-weight spiking neuron processing core (SNPC) for our 3D-NoC SNN processor, and the design of its on-chip learning block. The SNPC embeds 256 Leaky Integrate and Fire (LIF) neurons, and crossbar based synapses, covering a chip area of 0.12mm^2. Its performance is evaluated using MNIST dataset, achieving an inference accuracy of 97.55%.**

***Index Terms*—Spiking Neural Network, Neuromorphic, 3D Network on Chip, Spiking Neuron Processing Core**

I. Introduction

Spiking Neural Network (SNN) is the third generation of the computing paradigm "Neuro-inspired computing", modeled after the neural networks of the biological brain to best the setbacks of conventional computing (*Von Neumann* machines) [1]. This computing paradigm tries to model specifications that make the biological brain achieve rapid parallel computations in real-time, fault tolerance, power efficiency, and the ability to perform tasks like image recognition and language learning, that the conventional computing machine cannot [2]. Unlike its predecessors, the neurons in SNN also mimic the information transfer in biological neurons (event triggered), and this immensely enhances its performance [3]. Several SNN models exist, and their spiking behaviors are different because they abstract diverse features. Some of these models include; Integrate and Fire [4], Integrate and Fire with Adaptation [5], Quadratic Integrate and Fire [6], Fitshugh Nagumo [7], Morris Lecar [8], Hodgkin huxley [9], and Izhikevich [10].

Research in SNN has experienced some success over the years with most of its simulation in software, and this in turn, has brought about its application in tasks like image processing and character recognition [11]. However on the software platform, it is still faced with the architecture and power limitations of the *Von Neumann* architecture, which prevents its full potential from being utilized. To remedy this, hardware architectures (neuromorphic chips) that do not have the architectural limitation of the *Von Neumann* architecture, and aim at taking advantage of the sparsity of Spikes in SNN to reduce power, are being explored [12]. These architectures are based on spiking neurons which process input information only when they receive spike signals, and given a sparsely-distributed input spike train, the power consumption for such event-driven hardware would be reduced since large portions of the network that are not driven by current incoming spikes can be set into power gated mode.

Over the years, several neuromorphic chips have been designed and among them we have SpiNNaker (16 cores covering a chip area of 102mm^2) [13], TrueNorth (4,096 cores in a chip area of 430mm^2) [14], Neurogrid (one core im a chip area of 168mm^2) [15], BrainScaleS (352 cores , each covering a chip area of 50mm^2) [2], Loihi (128 cores covering a chip area of 60mm^2) [16], and MorphIC (4 cores covering a chip area of 2.86mm^2) [17]. However, the design of neuromorphic chips are not without challenges that need to be solved. Building in hardware such a spiking neuromorphic chip with massive number of synapses requires building small sized spiking neuro cores with low power consumption, efficient neurocoding scheme, light weight on chip learning and fault tolerance. Neuromorphic chips like Brainscale, SpiNNaker and Loihi embed on-chip learning, however none of them were implemented using 3D-NoC interconnect. Most neuromorphic SNNs are trained either with supervised learning algorithms based on back propagation, unsupervised algorithms based on Spike Time Dependent Plasticity (STDP) and its modifications, or converted from ANNs. The STDP learning algorithm

exemplifies the changing and shaping of connections between neurons (pre-synaptic and post-synaptic) in the biological brain with respect to time [18], and rely on the timing of pre- and post-synaptic spike to adapt the synaptic strength between the neurons [19].

In our previous work [20], we presented a 3D-NoC SNN processor architecture, an approach to attempt achieving the level of neuron density in the biological brain without trade off in power and footprint. Most existing neuromorphic chips are implemented on a 2D-NoC interconnect which faces power, footprint and communication challenges with increased number of components [21]. 3D-NoC on the other hand is an interconnect composed of multiple layers of 2D NoC with continuous vertical interconnect based on Through Silicon Vias (TSV) [22] and does not suffer the challenges of the 2D-NoC [23]. Our 3D-NoC SNN architecture [20] is a combination of 3D-NoC [24] and SNN processing elements. The SNN processing elements are considered as Spiking Neuron Processing Cores (SNPC), and the NoC topology attributes how the SNPCs are interconnected within the network. For communication within the scalable interconnection 3D network, spikes are sent as packets/flits and to ensure efficient communication, a fault tolerant multicast routing algorithm called Fault-Tolerant Shortest Path K-means based Multicast Routing(FTSP-KMCR) algorithm is presented [20] [25]. The focus however, was on the implementation and evaluation of the interconnect, and because the SNPC had not been implemented, the evaluation for hardware complexity in terms of area and power, and the evaluation for network performance in terms of latency, was done by injecting spikes into the network. This evaluation approach is not so efficient, therefore for more efficient evaluation, we present in this work the implementation of the Spiking Neuron Processing Core.

In this work, we present an implementation of the SNPC for the 3D-NoC SNN processor [20]. The SNPC architecture mainly comprises of 256 LIF neurons, a crossbar based synapse, and a control unit. we also show the design of the Spike Time Dependent Plasticity (STDP) learning block.

The rest of this paper is organized as follows: Section 2 describes the architecture of the 3D-NoC SNN processor, focusing on the interconnect and SNPC architecture. Section 3, presents the evaluation of the SNPC based on its performance on MNIST data set inference. Section 4 presents the Discussion, and section 5, the conclusion and future work.

II. Architecture and Implementation

A. Overall System Architecture

The 3D-NoC SNN architecture from our previous work [20] is a 3D mesh architecture composed of SNPCs and Fault Tolerant Multicast 3D Routers (FTMC-3DR) [20] [26], arranged in a 2D mesh topology on a tile. Multiple tiles are then stacked to form a 3D architecture. For design illustration, a 4x4 2D tile, stacked to form a 3D architecture was presented in [20]. However, it can be dimensioned in a manner suitable for the designer's application. Each SNPC embeds an adaptive array of LIF neurons with crossbar based synapse. Although with few modifications, a number of other neuron models can be supported. An output spike from a LIF neuron while performing a task is encoded and sent to the pre-synaptic neurons which can either be in the same SNPC or in another SNPC within the 3D network depending on the task mapping strategy. If in the same SNPC, the spike is received by the post-synaptic neurons after it has been identified, and the weight of its synapse with the pre-synaptic neuron is obtained from the synapse memory using the address deduced from the incoming spike in the crossbar. But if in another SNPC, the spike is packeted at the network interface (NI) and sent to the FTMC-3DRs that routes it to the destination SNPC where the post-synaptic neuron resides. At the destination SNPC, the packeted spike is decoded, the post-synaptic neuron identified, and together with the synpatic weight obtained through the crossbar, arrives the pre-synaptic neuron. The FTMC-3DR and the SNPC arechitecture are described in the following sections.

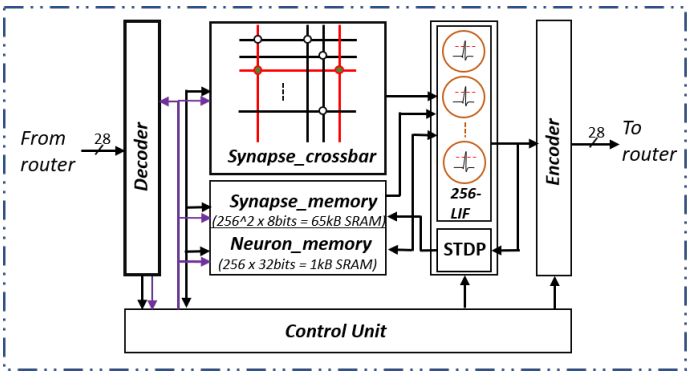

Fig. 1. Architecture of Spiking Neuron Processing Core (SNPC)

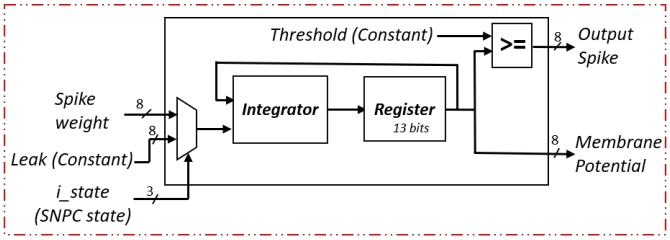

Fig. 2. Architecture Of A Single LIF Neuron

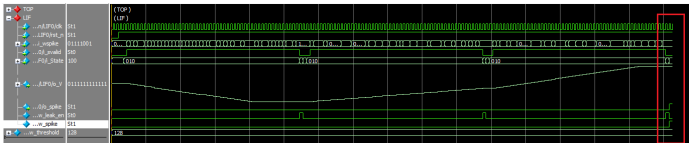

Fig. 3. Wave Form Of LIF Neuron Operation. The LIF neuron accumulates the weights received from the cross bar during the $Generate\_Spike\_\&\_Comp$ state of the control unit, and experiences leak during the $Leak$ state. At the $Fire$ state of the SNPC control unit, if the membrane potential has exceeded the threshold, an output spike is fired.

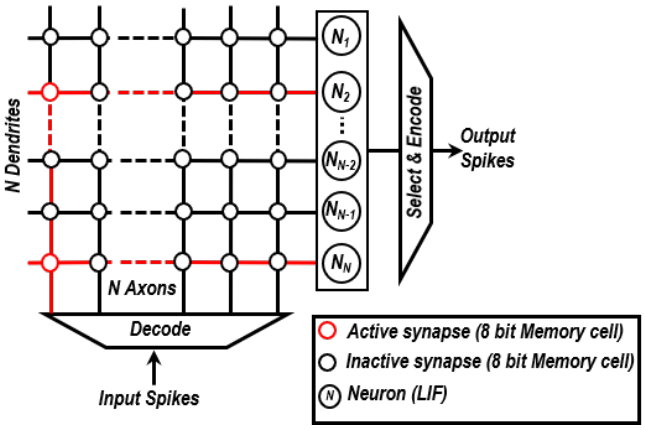

Fig. 4. Crossbar architecture, showing the axons, dendrites, synapses implemented with SRAM, and neurons. An incoming address event activates the first axon, causes its connections to neurons 2 and N to be read, and this in turn updates the neurons.

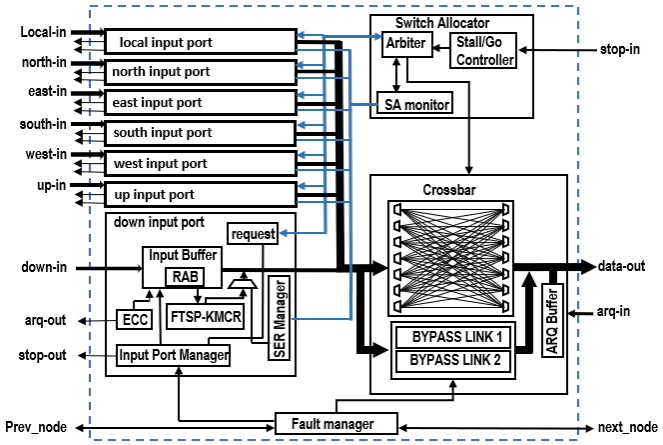

Fig. 5. Architecture Fault-tolerant Multicast Spike 3D Router architecture (FTMC-3DR) [20]

B. Interconnect architecture

Fig. 5 illustrates the architecture of the Fault Tolerant Multicast 3D Router (FTMC-3DR). Each SNPC in the architecture is accompanied by a FTMC-3DR with 7 inputs and 7 output ports, where 1 input and 1 output port is used to connect to the SNPC, 2 input and 2 output ports to the routers above and below, and the remaining 4 input and 4 output ports to the neighbouring routers, each in north, south, east and west direction. It is tasked with routing spikes across the network from source SNPC to destination SNPC using the Address Event Representation (AER) protocol. When an incoming packetized spike reaches the router, the router stores it in its input buffer, which is the first of its four pipelined stages; Buffer Writing (BW), Routing Calculation (RC), Switch Arbitration (SA), and Crossbar Traversal (CT) [27]. After entering the buffer, the packet is processed and the source address is obtained and calculated to arbitrate its output port. This is the second pipelined stage. After this is done, the switch arbiter grants permission in response to a request sent to it for an output port to be used, the third pipelined stage. Finally the packet is then sent to the proper output port through the crossbar in the router, this is the fourth pipelined stage. Detailed information on the FTMC-3DR can be found in our previous work [20].

C. Spiking Neuron Processing Core Architecture

The SNPC is the processing unit in the 3D-NoC SNN architecture. It is made up of a Controller, Crossbar, Synapse and Neuron memory, neuron array and STDP block. They are all described in the following sections. The SNPC architecture is illustrated in Fig. 1.

1) Control Unit: The control unit is tasked with managing the operations and state of the SNPC. It functions as a finite state machine, switching among seven states of SNPC operations; *Idle*, *Download_spike*, *Generate_Spike_&_Comp*, *Leak*, *Fire*, *Upload_spike* and *Learn*. At the *Idle* state, the SNPC does nothing, and from that state it transitions to the *Download_spike* state where it downloads incoming spikes in address event representation format to the crossbar. At the *Generate_Spike_&_Comp* state, the crossbar generates an SRAM address, and the weights stored in that address are fetched and sent to the corresponding LIF neuron for accumulation. The *Leak* state reduces membrane potential, the *Fire* checks the fire condition, the *Upload_spike* sends the output from the LIF array to other SNPCs, and the *Learn* state activates the STDP block for learning.

2) Crossbar: The neurons in the biological brain are arranged in a 3D manner, and this allows for more connection between them. To attempt having the same level of connection in the SNPC, a crossbar approach (which aims at merging memory and neuron update details) is used to implement the synapses. Fig. 4 shows a diagram of the crossbar which is a composite of an array of wires crossing each other in a rectangular manner with the horizontal representing the axon, and the vertical representing the dendrite of the LIF neurons. At the intersection of two wires is a memory cell, which stores the synaptic weight. To implement the N x N (N signifying the number of neurons) crossbar, an on-chip SRAM is used.

At the beginning of a simulation, the weights of the network are loaded into the synapse memory. Spikes are fed as vectors to the crossbar, so when a spike arrives the crossbar at a time-step, it is decoded to determine the address of the synapse weights. This address is fed as input to the synapse memory which then provides access to the synaptic weights stored at the given address. The weight is then read, and fed to the post-synaptic neuron for computation. Fig. 6 illustrates the pipelined process of fetching synaptic weights from synapse memory after its address has been decoded by the crossbar from an incoming spike.

3) Synapse Memory and Neuron Memory: The Synapse memory stores the synaptic weights, and Neuron memory store the data for neural computations. They are both implemented with SRAM.

4) LIF array: Fig. 2 shows the schematic diagram of the implemented LIF neuron which performs computations on the data read from the Crossbar and the memory. A LIF

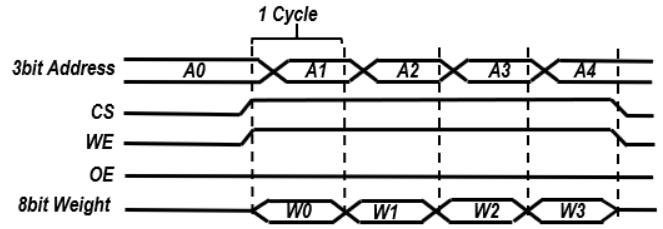

Fig. 6. Timing Diagram Of Loading Weights Into Synapse Memory.

neuron accumulates incoming spikes to increase its membrane potential while experiencing leakage. when the membrane potential crosses a threshold, it fires an output spike. This operation is illustrated in Fig. 7 and Fig 3

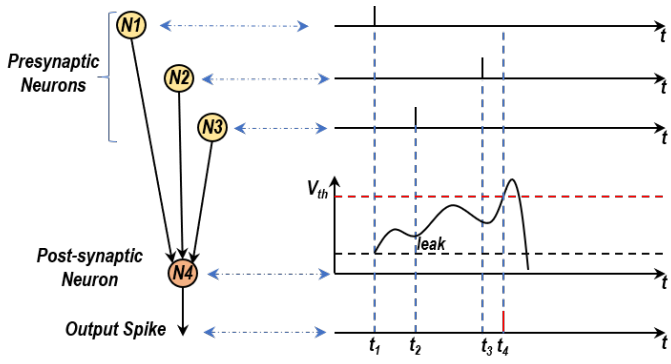

Fig. 7. LIF neuron operation: A post synaptic neuron (N4) receives spikes from presynaptic neurons (N1), (N2), (N3). The received spikes are accumulated, increasing the membrane potential of neuron (N4) from t_1 to t_3, while experiencing leak. At t_4, its membrane potential exceeds the threshold, an output spike is fired, and the membrane potential resets

Mathematically, the membrane potential V_j^l of a LIF neuron j in layer l at a time-step t is described as:

$$V_j^l(t) = V_j^l(t-1) + \sum_i w_{ij} * x_i^{l-1}(t-1) - \lambda \qquad (1)$$

where w_{ij} is the synaptic weight from neuron i to j, λ is the leak and x_i^{l-1} is pre-synaptic spike from previous layer $l-1$.

When the integration of input spikes is completed, the value of V_j^l at a time-step, is compared with the threshold value θ. If it exceeds, a spike is released and the neuron resets. This is mathematically expressed as:

$$\begin{cases} 1, \text{if } V_j^l > \theta \\ 0, \text{otherwise} \end{cases} \qquad (2)$$

5) STDP Block: At the learn state of the SNPC control unit, the control unit of the STDP block, *LB_control* is activated, and manages the learning process as a finite state machine in five states; *Idle*, *Accl*, *Before*, *After*, and *Done*. At the *Idle* state, the STDP block does nothing, and in the absence of a post-synaptic spike, jumps to *Done* state. In the event of a post-synaptic spike, 16 time steps and 16 pre-synaptc spikes are considered (8 before and 8 after the post synaptic spike). After every 16 time steps, the synaptic weights between 16 pre-synaptic neurons and the post synaptic neuron are updated. The *LB_control* moves to the *Accl* state, and spikes from the 16 pre-synaptic neurons are loaded from *Pre-Synp SRAM*. The memory address of their synaptic weights with the post-synaptic neuron is determined and the weights are fetched. At the *Before* state, the weights of the synapse between the 8 pre-synaptic neurons that fired before the post-synaptic neuron, are incremented. The *After* state on the other hand, decrements the weights of the synapse between the 8 pre-synaptic neurons that fired after and the post-synaptic neuron. At the *Done* state, a signal is sent to SNPC control unit and all the registers are cleared. A design of the STDP block is shown in Fig. 9, illustrating its method of weight update.

III. Evaluation

A. Evaluation methodology

In this section, we implement and evaluate the hardware complexity in terms of power and area of the SNPC without the STDP block , and its performance on MNIST dataset [28] inference accuracy. The SNPC was implemented in hardware, and the design was described using Verilog-HDL. The simulation and synthesis was carried out, using Cadence tools.

The SNN used to evaluate the performance of the implemented SNPC on MNIST dataset inference is a three layer fully connected SNN with 784 LIF neurons in the first layer, 225 in the second layer, and 10 in the third layer. The training approach adopted was to train an ANN, and then port the parameters of the pre-trained ANN to an equivalent SNN. Each 784 neuron in the first layer takes in 784 inputs, a 28×28 pixel grey scale image. Each image is first rescaled by dividing each pixel by 255 to keep the pixel range between 0 and 1, and then converted to spikes using poisson distribution [29]. Because the network is fully connected, each LIF neuron in the second and third layers receives 784 and 225 inputs respectively.The MNIST dataset [28] contains 60K training and 10K testing images, which are digits from 1 to 9. With this, we demonstrate the performance of the SNPC, and compare with existing works.

B. Evaluation result

In this section, we show the result of the hardware complexity and performance evaluation.

1) Hardware Complexity: The hardware complexity in terms of power and chip area of the SNPC design is given in TABLE I. The power report given is a sum of the leakage power, and the estimated dynamic power. In Table II, the chip area of several implementations is compared. *Merolla* [30] uses the highest hardware resource compared to other implementations. However, our implementation uses the smallest hardware resource. The SNPC layout is also shown ig Fig 10.

TABLE II presents a comparison of hardware complexity and performance on MNIST dataset.*Yin* [31] achieves the highest accuracy among the compared implementations followed by *Chen* [33], thanks to the size of their network. *Merolla* [30],

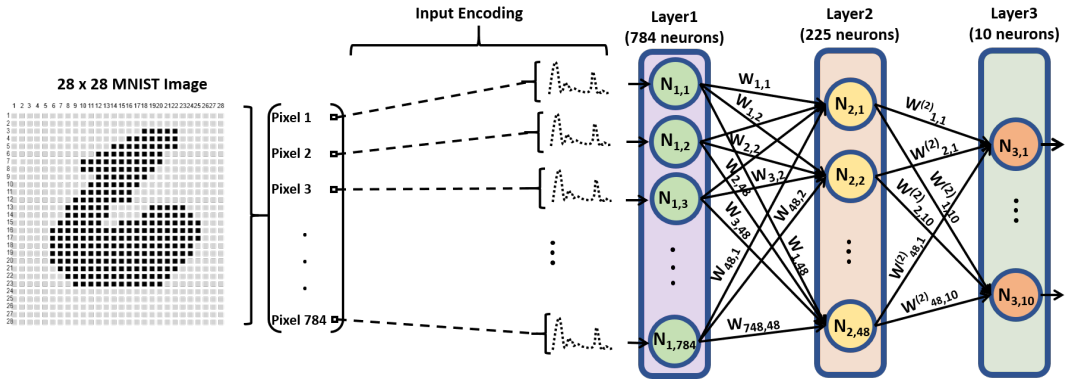

Fig. 8. Spiking Neural Network Architecture. The classified MNIST 28 × 28 gray scale images were first rescaled by dividing the value of each pixel by 255, and then encoded into spike arrays using poisson distribution.

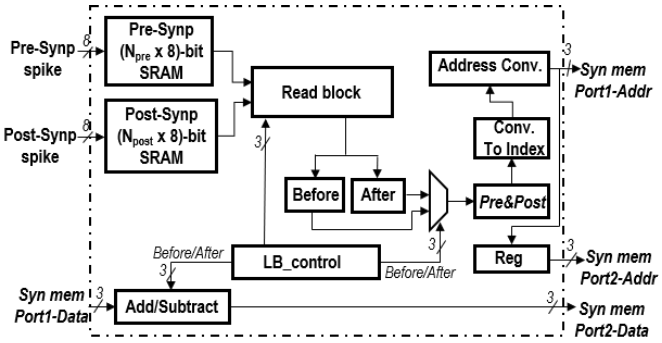

Fig. 9. STDP Block Architecture.

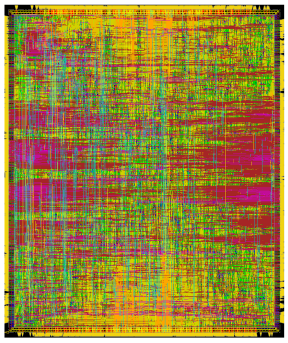

Fig. 10. SNPC Design Layout.

TABLE I
SNPC HARDWARE COMPLEXITY REPORT

System	SNPC
Power Estimation(mW)	493.5018
Area(mm^2)	0.12

SNPC and *Mostafa* [32] implements a smaller sized network. Nevertheless, *SNPC* achieves higher accuracy.

2) Performance: Using an SNN that was trained offchip, the SNPC was able to achieve inference accuracy of 97.55%.

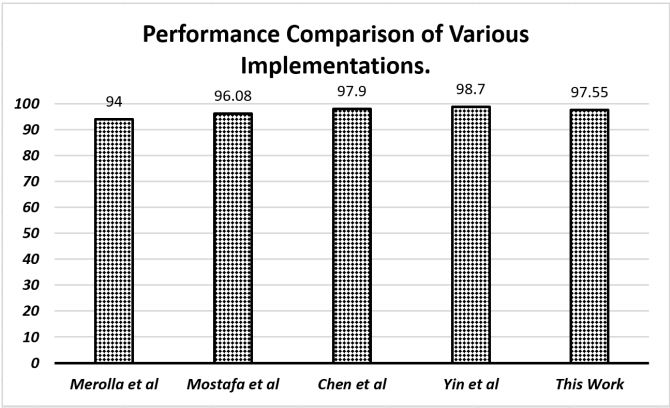

Fig. 11. Performance Comparison Over Various Learning Algorithms.

IV. DISCUSSION

In previous sections, we demonstrated SNPC, a processing element for our 3D-NoC SNN processor, and evaluated it for hardware complexity and performance. The target in general as stated previously is to design in hardware a neuromorphic chip with low power consumption, efficient neurocoding scheme, light weight on chip learning and fault tolerance. The implementation of the SNPC, is a step closer to this goal. Although a small sized network was used to evaluate the performance of the SNPC on MNIST image dataset classification, a larger network size can be used when it is integrated into the 3D-NoC SNN processor. This SNPC implementation houses 256 neurons and 64K synapses, but a point to note is that since the SNPC will be integrated into a 3D-NoC interconnect, each SNPC may not need to embed large number of neurons, but this depends on the requirement of the application.

TABLE II
SUMMARY OF AREA AND ACCURACY COMPARISON.

Parameters/Systems	Merolla et al [30]	Mostafa et al [32]	Chen et al [33]	Yin et al [31]	This Work
Core Area *(mm^2)*	4.2	-	1.72	1.65	0.12
Accuracy on MNIST (%)	94	96.08	97.9	98.7	97.55
Network Topology	2 Layers	3 Layers (FC)	4 Layers (FC)	3 Layers (FC)	3 Layers (FC)
Number of neurons	740	1394	2330	1306	1019
Implementation	Digital	Digital	Digital	Digital	Digital
Technology	45nm	FPGA	10nm FinFET	28nm	45nm
Learning	RBM	Backprop (off-chip)	Formal BNN (off-chip)	Backprop (off-chip)	(off-chip)

V. CONCLUSION AND FUTURE WORK

In this paper, we presented a digital implementation and evaluation of spiking neuron processing core (SNPC) for 3D-NoC SNN processor. A hardware complexity evaluation (for area and power), and a performance evaluation (accuracy on MNIST image dataset classification) was done. When compared with previously proposed implementation,the evaluation results show that SNPC provides a good trade-off between area and accuracy. Future work will explore learning approach on the SNPC, and its integration into the 3D-NoC SNN processor. The hardware complexity of the overall system will be evaluated, and for performance, several other applications that will take advantage of the 3D-NoC interconnect will also be explored. Other neuron models and learning algorithms could also be considered, depending on the application.

REFERENCES

[1] T. Wunderlich, A. F. Kungl, E. Müller, A. Hartel, Y. Stradmann, S. A. Aamir, A. Grübl, A. Heimbrecht, K. Schreiber, D. Stöckel, C. Pehle, S. Billaudelle, G. Kiene, C. Mauch, J. Schemmel, K. Meier, and M. A. Petrovici, "Demonstrating advantages of neuromorphic computation: A pilot study," *Frontiers in Neuroscience*, vol. 13, p. 260, 2019. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2019.00260

[2] G. Indiveri, B. Linares-Barranco, T. J. Hamilton, A. van Schaik, R. Etienne-Cummings, T. Delbruck, S.-C. Liu, P. Dudek, P. Häfliger, S. Renaud, J. Schemmel, G. Cauwenberghs, J. Arthur, K. Hynna, F. Folowosele, S. Saighi, T. Serrano-Gotarredona, J. Wijekoon, Y. Wang, and K. Boahen, "Neuromorphic silicon neuron circuits," *Frontiers in Neuroscience*, vol. 5, 2011.

[3] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge University Press, 2014.

[4] P. Dayan and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. The MIT Press, 2001.

[5] W. Gerstner, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002.

[6] P. E. Latham, B. J. Richmond, P. G. Nelson, and S. Nirenberg, "Intrinsic dynamics in neuronal networks. i. theory," *Journal of Neurophysiology*, vol. 83, no. 2, pp. 808–827, Feb. 2000.

[7] W. E. Sherwood, "Fitzhughnagumo model," in *Encyclopedia of Computational Neuroscience*. Springer New York, 2014, pp. 1–11.

[8] B. A., *Encyclopedia of Computational Neuroscience*. Springer, 2015.

[9] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *Bulletin of Mathematical Biology*, vol. 52, no. 1, pp. 25–71, Jan 1990.

[10] E. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003.

[11] B. Meftah, O. Lézoray, S. Chaturvedi, A. A. Khurshid, and A. Benyettou, *Image Processing with Spiking Neuron Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 525–544.

[12] M. Bouvier, A. Valentian, T. Mesquida, F. Rummens, M. Reyboz, E. Vianello, and E. Beigne, "Spiking neural networks hardware implementations and challenges: A survey," *J. Emerg. Technol. Comput. Syst.*, vol. 15, no. 2, pp. 22:1–22:35, Apr. 2019. [Online]. Available: http://doi.acm.org/10.1145/3304103

[13] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, May 2014.

[14] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, Aug. 2014.

[15] J. Geddes, S. Lloyd, A. Simpson, M. Rossor, N. Fox, D. Hill, J. V. Hajnal, S. Lawrie, A. Mclntosh, E. Johnstone, J. Wardlaw, D. Perry, R. Procter, P. Bath, and E. Bullmore, "Neurogrid: using grid technology to advance neuroscience," in *18th IEEE Symposium on Computer-Based Medical Systems (CBMS'05)*, June 2005, pp. 570–572.

[16] M. Davies, N. Srinivasa, T. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, Y. Liao, C. Lin, A. Lines, R. Liu, D. Mathaikutty, S. McCoy, A. Paul, J. Tse, G. Venkataramanan, Y. Weng, A. Wild, Y. Yang, and H. Wang, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, January 2018.

[17] C. Frenkel, J. Legat, and D. Bol, "Morphic: A 65-nm 738k-synapse/mm^2 quad-core binary-weight digital neuromorphic processor with stochastic spike-driven online learning," *IEEE Transactions on Biomedical Circuits and Systems*, pp. 1–1, 2019.

[18] D. Hebb, *The Organization of Behavior: A Neuropsychological Theory*. Psychology Press, 2002.

[19] L. Chen, C. Li, T. Huang, X. He, H. Li, and Y. Chen, "Stdp learning rule based on memristor with stdp property," in *2014 International Joint Conference on Neural Networks (IJCNN)*, July 2014, pp. 1–6.

[20] T. H. Vu, O. M. Ikechukwu, and A. Ben Abdallah, "Fault-tolerant spike routing algorithm and architecture for three dimensional noc-based neuromorphic systems," *IEEE Access*, vol. 7, pp. 90 436–90 452, 2019.

[21] Y. Ghidini, T. Webber, E. Moreno, I. Quadros, R. Fagundes, and C. Marcon, "Topological impact on latency and throughput: 2d versus 3d noc comparison," in *2012 25th Symposium on Integrated Circuits and Systems Design (SBCCI)*, Aug 2012, pp. 1–6.

[22] K. N. Dang, M. Meyer, Y. Okuyama, and A. B. Abdallah, "A low-overhead soft–hard fault-tolerant architecture, design and management scheme for reliable high-performance many-core 3d-NoC systems," *The Journal of Supercomputing*, vol. 73, no. 6, pp. 2705–2729, Jan. 2017. [Online]. Available: https://doi.org/10.1007/s11227-016-1951-0

[23] A. Milenkovic and V. Milutinovic, "A quantitative analysis of wiring lengths in 2d and 3d vlsi implementation of 2d systolic arrays," in *1997 21st International Conference on Microelectronics. Proceedings*, vol. 2, Sep. 1997, pp. 833–836 vol.2.

[24] K. N. Dang, A. B. Ahmed, Y. Okuyama, and B. A. Abderazek, "Scalable design methodology and online algorithm for tsv-cluster defects recovery in highly reliable 3d-noc systems," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2017.

[25] T. H. Vu, Y. Okuyama, and A. B. Abdallah, "Comprehensive analytic performance assessment and k-means based multicast routing algorithm and architecture for 3d-noc of spiking neurons," *J. Emerg. Technol. Comput. Syst.*, vol. 15, no. 4, pp. 34:1–34:28, Oct. 2019. [Online]. Available: http://doi.acm.org/10.1145/3340963

[26] H.-T. Vu, Y. Okuyama, and A. Ben Abdallah, "Analytical performance assessment and high-throughput low-latency spike routing algorithm for spiking neural network systems," *The Journal of Supercomputing*, vol. 75, 03 2019.

[27] K. N. Dang, A. B. Ahmed, X.-T. Tran, Y. Okuyama, and A. B. Abdallah, "A comprehensive reliability assessment of fault-resilient network-on-chip using analytical model," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 11, pp. 3099–3112, Nov. 2017. [Online]. Available: https://doi.org/10.1109/tvlsi.2017.2736004

[28] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[29] M. Fatahi, M. Ahmadi, M. Shahsavari, A. Ahmadi, and P. Devienne, "evt_mnist: A spike based version of traditional MNIST," *CoRR*, vol. abs/1604.06751, 2016. [Online]. Available: http://arxiv.org/abs/1604.06751

[30] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. S. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm," in *2011 IEEE Custom Integrated Circuits Conference (CICC)*, Sep. 2011, pp. 1–4.

[31] S. Yin, S. Venkataramanaiah, G. Chen, R. Krishnamurthy, Y. Cao, C. Chakrabarti, and J. sun Seo, "Algorithm and hardware design of discrete-time spiking neural networks based on back propagation with binary activations," in *2017 IEEE Biomedical Circuits and Systems Conference, BioCAS 2017 - Proceedings*, vol. 2018-January. Institute of Electrical and Electronics Engineers Inc., 3 2018, pp. 1–4.

[32] H. Mostafa, B. U. Pedroni, S. Sheik, and G. Cauwenberghs, "Fast classification using sparsely active spiking networks," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–4.

[33] G. Chen, R. Kumar, H. Sumbul, P. Knag, and R. Krishnamurthy, "A 4096-neuron 1m-synapse 3.8pj/sop spiking neural network with on-chip stdp learning and sparse weights in 10nm finfet cmos," 06 2018, pp. 255–256.