IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

On the Design of a Fault-tolerant Scalable Three Dimensional NoC-based Digital Neuromorphic System with On-chip Learning

OGBODO MARK IKECHUKWU[1], (Student Member, IEEE), KHANH N. DANG[1,2],(MEMBER, IEEE), AND ABDERAZEK BEN ABDALLAH[1], (Senior Member, IEEE)

[1]Adaptive Systems Laboratory, Graduate School of Computer Science and Engineering, The University of Aizu, Aizu-Wakamatsu, Fukushima 965-8580, Japan
[2]SISLAB, University of Engineering and Technology, Vietnam National University Hanoi, Hanoi, 123106, Vietnam

Corresponding author: Ogbodo Mark Ikechukwu (e-mail: d8211104@u-aizu.ac.jp); Abderazek Ben Abdallah (e-mail: benab@u-aizu.ac.jp)

ABSTRACT Neuromorphic systems have shown improvements over the years, leveraging Spiking neural network's (SNN) event-driven nature to demonstrate low power consumption. As neuromorphic systems require high integration to form a functional silicon brain-like, moving to 3D integrated circuits (3D-ICs) with three-dimensional network on chip (3D-NoC) interconnect is a suitable approach that allows scalable design, shorter connections, and lower power consumption. However, highly dense neuromorphic systems also encounter the reliability issue where a single point of failure can affect the systems'operation. Because neuromorphic systems rely heavily on spike communication, an interruption or violation in the timing of spike communication can adversely affect the performance and accuracy of a neuromorphic system. This paper presents NASH[a], a fault-tolerant 3D-NoC based neuromorphic system that incorporates as processing elements, lightweight spiking neuron processing cores (SNPCs) with spike-timing-dependent-plasticity (STDP) on-chip learning. Each SNPC houses 256 leaky integrate-and-fire (LIF) neurons and 65k synapses. Evaluation results on MNIST classification, using the fault-tolerant shortest-path K-means-based multicast routing algorithm (FTSP-KMCR), show that the NASH system can maintain high accuracy for up to 30% permanent fault in the interconnect with an acceptable area and power overheads when compared to other existing systems.

[a]This project is supported by the University of Aizu, Competitive Research Funding (CRF), Ref. UoA-P6-2020

INDEX TERMS Digital Neuromorphic, Spiking Neural Network, Fault-tolerant, 3D-NoC, Architecture

I. INTRODUCTION

SPIKE based neuro-inspired computing has gradually gained awareness both to provide better insight into the brain's computation and to explore this computation through event-driven neuro-inspired systems. Spiking neural networks (SNN) are used in many applications, such as vision systems [1], brain-computer interface [2], neuro-robotics control and decision making [3], and brain simulation [4]. SNN tries to simulate the information processing in the biological brain, which requires parallel arrays of neurons to communicate via spikes. In contrast to multi-layer perceptrons where all neurons fire at every propagation cycle, SNN fires only when its voltage potential is stimulated beyond a threshold value [5]. In encoding information as spikes, SNN employs a coding scheme which could be rate coding, population coding, or temporal coding [6].

Several spiking neuron models exist, and the prevalent ones which are often found in typical SNNs qre the integrate and fire (IF) model [7] and its variants which include the leaky integrate and fire (LIF). The neuronal dynamics of this model can be thought of as an integration process, together with a spiking mechanism. Typical spikes irrespective of their amplitude and shape are handled as similar events, and from the outset to finish, lasts about two milliseconds [8] traveling down axonal lengths. Another spiking neuron model which is noted for its detailed simulation of the ion channels of a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

biological neuron is the Hodgkin, and Huxley model [9]. This model is nonlinear and stochastic. However, it is complex, making it less ideal for large-scale simulation and hardware implementation.

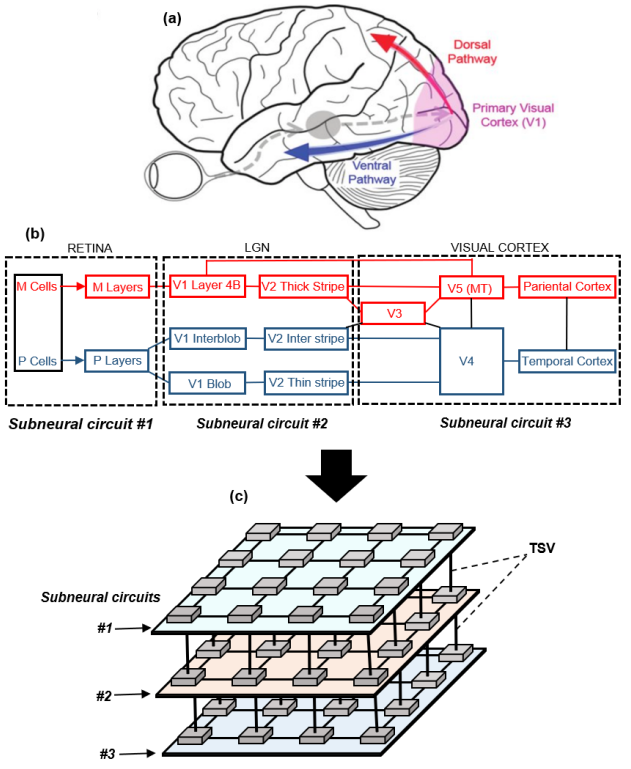


FIGURE 1: An illustration of large scale neural task mapping on a 3D neuromorphic architecture. (a) Illustration of information flow path from retina to primary visual cortex (V1) in the human brain. (b) Block diagram of connections among key neural structures in the human vision system. (c) 3D neuromorphic architecture with through silicon vias (TSVs) for vertical inter-layer connection capable of supporting the high level of connectivity required for the vision task mapped according to the partitions in (b). This image is inspired by the work in [10].

Software simulation of SNN [11] [12] has shown to be a flexible way of exploring the behavior of neuronal systems. However, simulating deep SNN in software is slow and consumes much power, making it less suitable for both implementing real-time systems and precise large-scale simulations of neural systems. Hardware implementation (neuromorphic system) on the other hand, provides an alternative approach that holds the potential for independent spikes to be generated accurately, and output spikes to be simultaneously fired in real-time. Also, hardware implementation has the edge of computational acceleration over software simulations, and peculiar multi-neuro-core neuromorphic architectures can leverage the parallelism and spike sparsity available in SNN to deliver rapid processing with low power. Recent artificial neural networks (ANNs) consist of several layers, and this has brought about the term deep neural network (DNN). Each layer of a DNN is often expressed as a two dimensional (2D) structure, making the entire network itself a three dimensional (3D) structure. Mapping such 3D structure onto a 2D circuit usually leads to several lengthy wires between layers or congestion points [10], [13].

To build in hardware an SNN architecture with large number of synapses, there are several hurdles that need to be surmounted. First, there is a need for a densely parallel architecture with low-power consumption, light-weight spiking neuro processing cores (SNPCs) with on-chip learning, and efficient neuro-coding scheme. Another major hurdle that needs to be surmounted is the on-chip neurons communication and routing. Furthermore, we need to consider that the number of neurons to be connected are magnitudes of times larger than the number of cores that need to be interconnected on recent multicore system on chip platforms [14]. These hurdles make the building of such a neuromorphic integrated circuit (IC) a challenging task [15].

A. BACKGROUND AND MOTIVATION

Recent progress in tract-tracing connectomics has helped deepen our understanding of the topology of the brain [23] [24] and has buttressed the findings that the anatomical topology of the brain network is organized as a three-dimensional small world network. Such a network is typified by dense local clustering of neurons with short connection lengths, and a few long-range connections between clusters [25], [26]. Therefore, the brain connectivity can generally be described at three levels of scale: first, the single synaptic connections that link individual neurons at the micro-scale. Second, the networks connecting neuronal populations at the mesoscale, and third, brain regions linked by fiber pathways at the macro-scale [27]. In representing these connections in neuromorphic systems, a crossbar is one of the approaches employed. However, it has been demonstrated that the size of a crossbar directly affects the power consumption of a neuromorphic system [28] [29]. This limitation has been observed in neuromorphic systems that employed single large crossbars [30]. Therefore, for a scalable neuromorphic system that supports large SNN with massive number of synapses, a partitioning and mapping of its synapses into smaller local crossbars which are linked using a shared interconnect is a better approach. Unfortunately, with the use of shared interconnect, the challenge of latency which affects the timing of spikes, is introduced.

In SNN, the timing of synaptic spikes play an important role in the proper functioning of the network [10]. The timing of a postsynaptic spike is completely influenced by the arrival time of pre-synaptic spikes [31]. Therefore, any violation of this timing will negatively affect the operation of the spiking neurons. Also, since there is a high level of local communication among neurons, incoming spikes are distributed among neighboring neurons.

There are various communication mediums that are used

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

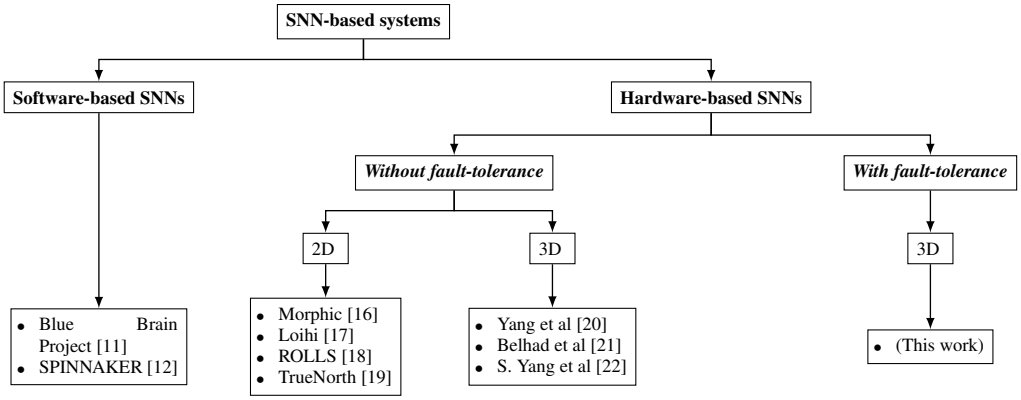


FIGURE 2: Classification of SNN based systems based on platform, architecture and fault-tolerance.

when designing an interconnect, and they include shared bus and packet-switched network on chip (NoC) [32]. A shared bus is a poor choice when implementing a large-scale SNN with multicast routing since it suffers adversely with increased number of nodes. The nonlinear increase in neural connectivity will be too much for such an interconnect to handle. An interconnect that has been considered as a potential solution is the 2D packet-switched NoC (2D-NoC). However, with further scaling, 2D-NoC interconnects begins to experience communication challenges that affect power and performance, especially in large-scale SNN chips. 3D packet-switched NoC (3D-NoC), on the other hand, enables scaling and parallelism in the third dimension by combining NoC and 3D ICs (3D-ICs) [33]. With the help of its short through silicon vias (TSVs) that enable communication between layers, it can reduce communication costs. These merits of 3D-NoC make it suitable for large-scale SNN applications. Moreover, the brain is biologically organized in a 3D structure; therefore, by adopting the 3D interconnect as illustrated in Fig. 1, neuromorphic systems can inherit the shape, and the interconnects of a biological brain.

Despite being known for having some underlying fault-tolerance attribute resulting from their densely parallel framework, SNNs face some fault challenges, especially those assumed from implementing them in hardware [34]. To handle some of these faults, a fault-tolerant approach needs to be considered.

In this paper, we present the architecture, design and preliminary evaluation of a fault-tolerant scalable 3D-NoC-based neuromorphic system with on-chip learning (NASH). To the best of our knowledge, this is the first implementation of a fault-tolerant neuromorphic system based on a 3D NoC. The rest of this paper is organized as follows: Section II presents a review of prior works akin to this research area. Section III describes the architecture of the system's main building blocks. In section IV, we give the evaluation results. The last section gives the conclusion and future work.

II. PRIOR WORKS

Quite a number of neuromorphic processors have been proposed for applications ranging from brain simulation [4] to automotive sensors [35]. These applications determine the design approach and components of neuromorphic systems. A taxonomy of the well known related works is presented in Fig. 2.

One of the neuromorphic chips in 2D domain is Loihi [17], a 14-nm 60-mm^2 manycore on-chip learning processor. The processor uses off-chip communication interfaces in a hierarchical mesh fashion to connect chips. In [16], a quad-core binary weight neuromorphic chip was proposed. Each core houses 512 LIF neurons, a total of 528K binary synapses grouped in three levels, and a stochastic spike-driven synaptic plasticity learning. Besides the fully integrated digital SNN chips mentioned above, mixed-signal works are also attractive since they can directly emulate the dynamics of real neural systems. Consequently, they benefit from the emulation of biological neural computing systems. One of the examples for this type is the ROLLS neuromorphic processor [18]. ROLLS has 128K analog synapses and 256 neuron circuits that support biologically plausible dynamics and bi-stable spike-based plasticity mechanisms that endow it with on-line learning abilities. Besides, asynchronous digital logic circuits are also integrated for synapse, neuron, and network configurations. ROLLS was evaluated, and it showed good characteristics in the emulation of computational neuroscience models and a pattern recognition task. However, it requires a complicated implementation to deal with challenges of scalability and precision. TrueNorth is another 2D neuromorphic chip with 4096 cores arranged in a mesh topology of 64×64. Each core houses 65K synapses, and with time division multiplexing, uses a single shared neuron circuit to compute the state of 256 neurons.

Apart from 2D neuromorphic implementations, 3D ones take full advantage of 3D-ICs. The work in [21] investigated the architecture and design of a 3D stacked neuromorphic accelerator. The 3D stacking architecture used face-to-face bonding of two 20cm wafers with micro-bumps. A Recent work was presented in [20] about a neuromorphic system

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

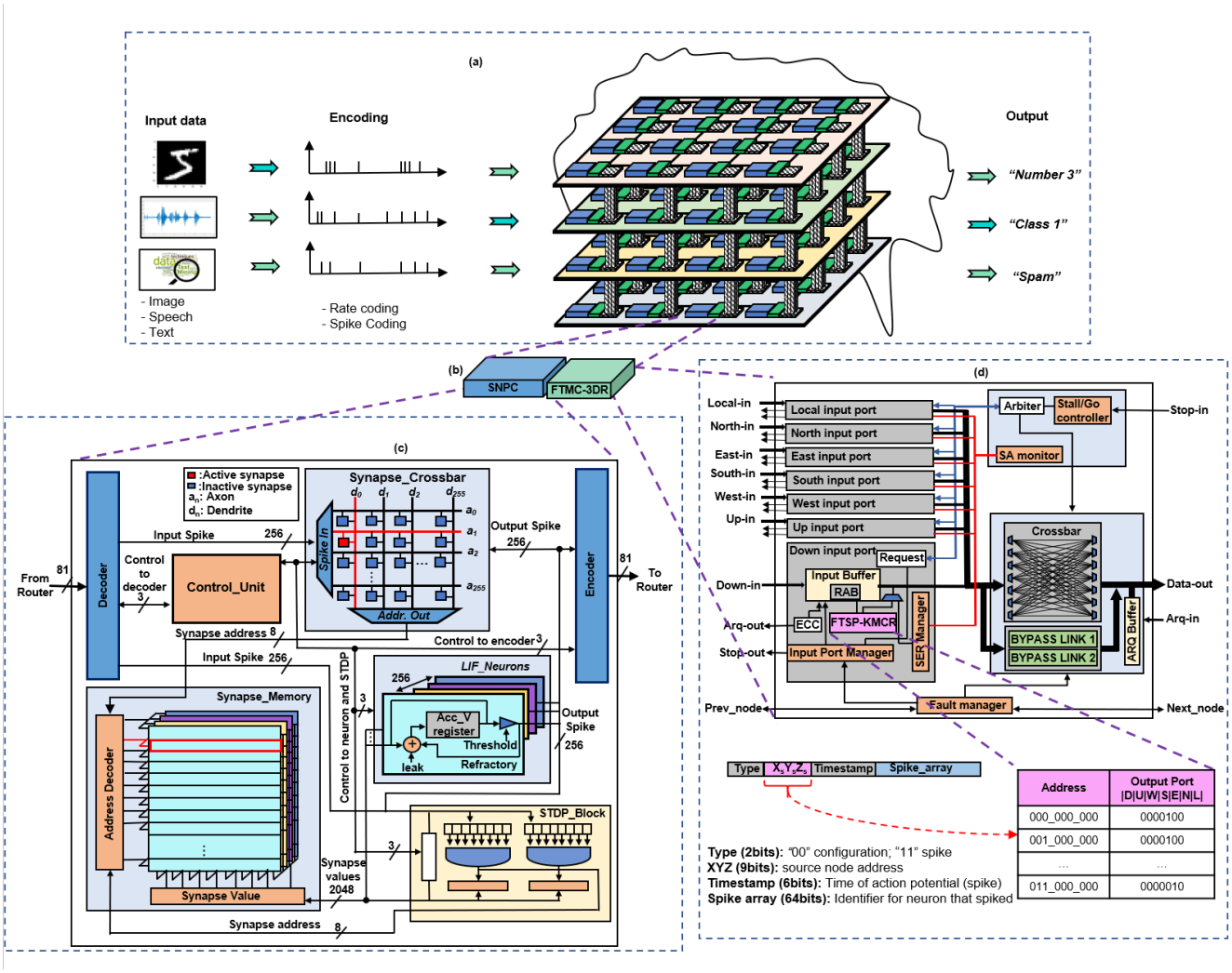


FIGURE 3: High level view of NASH System Architecture: (a) NASH system architecture illustrated in a $4\times4\times4$ configuration. (b) A single node in NASH system comprising of an SNPC and a FTMC-3DR. (c) An architecture of the SNPC comprising of the synapse memory, synapse crossbar, LIF neurons, control unit, encoder/decoder, and an STDP learning block (d) Architecture of the FTMC-3DR comprising of the input ports, crossbar, swich allocator, arbiter and flow control.

for the simulation of large-scale conductance-based SNNs. The architecture was implemented in six Altera Stratix III FPGA boards to simulate one million neurons [20]. An *AER* multicast routing mechanism was used for inter-neuron communications. Although the NoC architecture meets the requirements of the system, it is hardly deployed in embedded neuromorphic systems [36]. The authors in [22] proposed a neuromorphic architecture for large-scale biophysically meaningful neural networks with multi-compartment neurons in a 3D-NoC architecture, using four Altera Stratix III FPGAs. However, this system utilizes a butterfly fat-tree-based topology, which although mimics the brain's connectivity more closely, cannot be mapped on an ASIC design as its shape will lead to wastage of silicon area. Moreover, the point-to-point connection at the parent node limits path diversity, and due to the bandwidth imbalance in this topology, thermal hot-spots can be created. In conventional 3D-ICs, TSVs are used as vertical links between layers, enabling a reduction in the number of hops compared to 2D counterparts [37]. In addition to that role, TSVs enable mimicking biological characteristics of spiking neurons, as shown in [38], where the authors proposed a new method to use TSVs as neuronal membrane and how to enhance the TSV capacitance significantly.

III. NASH SYSTEM ARCHITECTURE

Figure 3 shows the overall architecture of the proposed fault-tolerant 3D-NoC-based neuromorphic system (NASH). The system is a 3D mesh-based architecture composed of several nodes described in Fig. 3 b, c and d. Each node is made up of a spiking neuron processing core (SNPC), a fault-tolerant multicast 3D router (FTMC-3DR) and a network interface (NI) (not shown in Fig. 3 for simplicity). Fig. 3a shows a $4 \times 4 \times 4$ NASH configuration. Each SNPC embeds 256

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

LIF neurons with crossbar-based synapse. The LIF model is adopted in this work because it has proved to be effective for most learning applications and is suitable for digital implementation due to its modest hardware cost [39]. An output spike from a LIF neuron while performing a task is sent to post-synaptic neurons, which could either be in the same SNPC or in another within the 3D network. If in the same SNPC, the spike is received by the post-synaptic neurons after it has been identified, and the weights of its synapse with the pre-synaptic neurons is obtained through the crossbar. However, if the destination SNPC is different, the spike is encoded at the network interface as a packet and sent to the local FTMC-3DR, which routes it to the destination SNPC(s) where the post-synaptic neurons reside. At the destination SNPC(s), the spike is decoded, the post-synaptic neurons identified, and together with the synapse value obtained through the crossbar, they are sent to the post-synaptic neurons.

Figure 3d illustrates the architecture of the FTMC-3DR (fault-tolerant multicast 3D router), which is based on [10], [40]. The FTMC-3DR has 7 input (output) ports: one port for connecting to the local SNPC with which spike packets can be injected into or received from the network, 4 for connecting to neighboring routers in the north, east, south, west direction using the intra-layer links, and the remaining up and down ports for those in the closest layers through the TSVs. Each FTMC-3DR routes multicast packets in four pipeline stages. In the first stage, which is buffer writing (BW), the received spike packets are stored in the port's input buffer. In the second pipeline stage, routing calculation (RC) obtains the source address of the stored packet and uses it to derive the next address, which is either in the X, Y, or Z dimension. After this address is derived, the switch-allocator (SA), which handles a stall/go flow-control, and the matrix arbitration (matrix-arbiter scheduler) are triggered in the third stage to allocate the right port to the next router or local SNPC. After the right output port has been allocated, the fourth stage, crossbar traversal (CT) begins, and the packet traverses the crossbar to the allocated output port.

A. SPIKING NEURON PROCESSING CORE (SNPC)

The SNPC is shown in Fig. 3c, and it contains 256 physical LIF neurons, a crossbar-based synapse, a control unit, a synapse memory, and an STDP learning module. The SNPC design enables the state of neurons to be multiplexed onto a single 256-bit shared bus, each bit taking the value of 1 or 0 to signify the presence or absence of a spike event, respectively. The block diagram of the implemented LIF neuron model is shown in Fig. 4. The neuron membrane voltage is accumulated by adding up the input synapse values in the integrator. The resulting value is then stored in the 14-bit register, which utilizes 13-bit to store the membrane voltage value and 1-bit for overflow. To mimic the leak current found in the neural membrane, a set leak value that causes decay in the membrane voltage value is received by the neuron when the leak is activated. When the value of the accumulated membrane voltage exceeds the set threshold, a 1-bit output spike is fired, and a signal is sent to the register to reset the value of the membrane voltage to zero and begin the refractory period. While in refractory, the neuron does not accumulate synapse values. As illustrated in Fig. 5, the refractory count gradually counts down every time step from the set refractory period to 0, and afterward, the neuron can accumulate synapse values again. The synapse crossbar architecture in the SNPC is

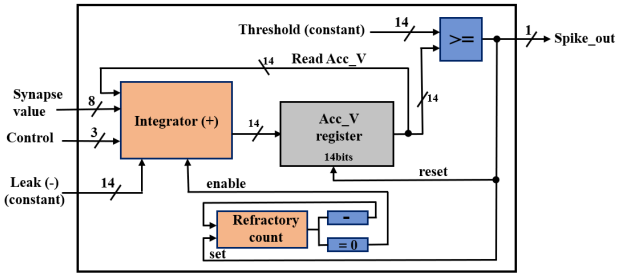


FIGURE 4: Block Diagram of LIF Neuron. It's operation while performing integration of synapse values, leak, and fire are in response to the control unit state signal.

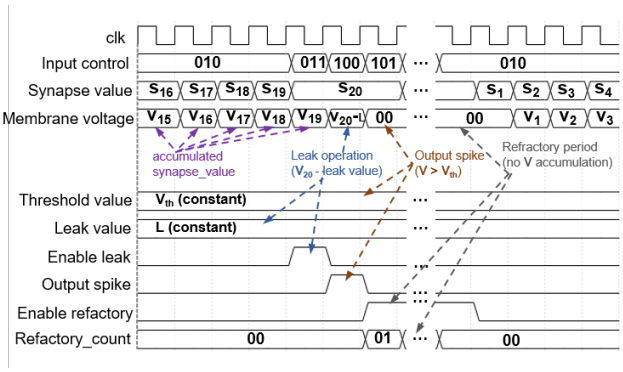


FIGURE 5: Operation diagram of a LIF neuron leading to the accumulation of synapse values that increases the membrane voltage, the leakage of membrane voltage, the release of output spikes as the membrane voltage exceeds the threshold value, and the refractory that follows afterwards.

described in Fig. 3c, and represents a total of 65k synapses for all 256 embedded neurons. The control unit of the SNPC controls the neuron update operation of the crossbar. As described in Fig. 6, when an input spike array is received at the crossbar, it is checked for the presence of spike event(s). In the presence of spike event(s), one hot mechanism is applied to get the memory address of the associated synapses whose values are stored on the on-chip SRAMs. The synapse values stored at these memory addresses are then fetched from the synapse memory and sent to the corresponding post-synaptic neurons for update. In the absence of spike events, the neuron update does not occur, and the SNPC moves to the next operation. Leveraging the SNPC's architecture, the synapse crossbar can perform parallel neuron update, enabling the entire 256 neurons to be updated in one cycle.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

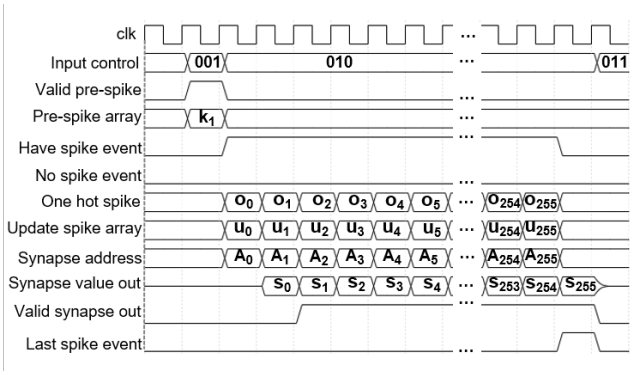


FIGURE 6: Illustration of neuron update operation at the crossbar. An input presynaptic spike array is stored and checked for spike events. If present, the *Have spike event* signal becomes high. Afterwards, the one hot operation to get the synapse address begins, updating the one hot spike array for every spike event: from O_0 to O_{255}. The stored presynaptic spike array is also updated after each spike event is processed: from U_0 to U_{255}. The synapse address is then used to fetch the synapse values from the synapse memory, and sent to the postsynaptic neurons. When the last spike event in the array has been processed, the crossbar sends a signal to the control unit signaling that all spike events have been processed.

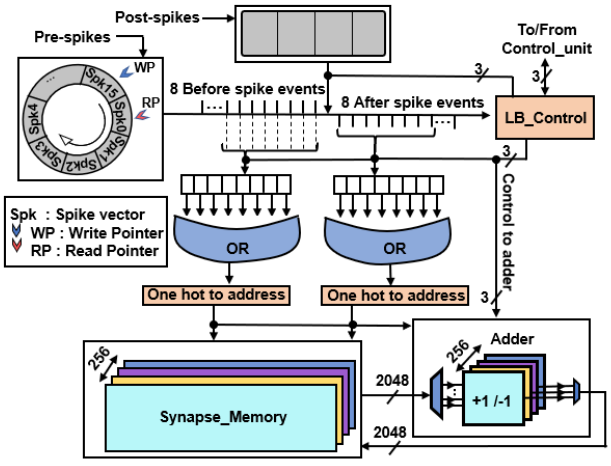


FIGURE 7: STDP Learning Module: The trace based STDP learning considers 16 pre-synaptic spikes, 8 before and 8 after in relation to post synaptic spike(s) before enabling synaptic update. Because synapse update does not happen at every spike, energy and learning time is reduced.

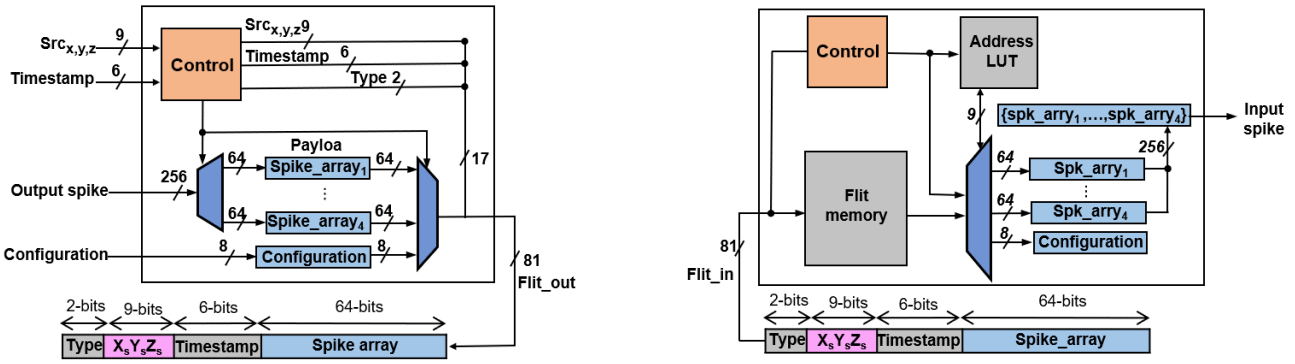
To achieve on-chip learning in each of the 65k synapses represented in the synapse crossbar, an efficient implementation of the trace-based STDP learning rule [41] is performed. The update logic of the implemented trace-based STDP is presented in Fig. 7. A learning operation requires 16 pre-synaptic spike trace arrays, each from a classification time step. These pre-synaptic spike trace arrays are stored in a circular memory using a 4-bit time step counter. To begin learning, the presence of a postsynaptic spike trace array(s) stored in a memory is verified. Then the pre-synaptic spike trace arrays are grouped into 8 $Before$ and 8 $After$, based on their arrival time relative to the postsynaptic spike trace array(s). An OR operation is further performed on the 8 $Before$ spike arrays and on the 8 $After$ spike arrays to obtain two arrays. Using one hot operation and the postsynaptic spike trace array(s), the associated synapses' memory addresses are obtained from the $Before$ and $After$ arrays. The corresponding synapse values are then fetched from the synapse memory, increased for the $Before$ spike events, decreased for the $After$ spike events, and then written back to the synapse memory. The implemented trace-based STDP enables the parallel update of synapses.

The control unit of the SNPC is designed as a state machine, controlling the SNPC's operation. It starts off Idle, which is the first and default state. At this state the SNPC does nothing while waiting for input spikes. The arrival of a presynaptic spike array is preceded by a signal which triggers the control unit to change its state to the second state allowing the presynaptic spike array to be received. After the presynaptic spike array is received, the third state is enabled. This state activates the crossbar to identify and update the postsynaptic neurons by sending the corresponding synapse values stored in the synapse memory. After the last postsynaptic neuron has been updated, a signal is sent from the crossbar to the control unit to move to the fourth state. The fourth state enables decay in the value of the postsynaptic neurons' membrane voltage, and is followed by a fifth state that triggers the postsynaptic neurons to check for an output spike by comparing the value of their membrane voltage with that of the set threshold. At the sixth state, the spikes generated by the neurons are sent to the network interface for encoding, after which they are sent to the destination SNPC. When the sixth state is complete, the condition for learning is examined. If satisfied, the learning, which is the seventh state, begins, and when finished, a signal is sent to the control unit to return to the default state. If the learning condition is not satisfied, the learning is skipped to the default state.

B. FAULT-TOLERANT NEURONS INTERCONNECT

The encoder and decoder in Fig. 3c serves as the NI between an SNPC and its local FTMC-3DR. As described in Fig. 8, the encoder is used to encode spikes into packets for transmission. In contrast, the decoder is used to decode received packets to spikes. The NI encodes spikes using an 81-bit flit format shown in Fig. 3d. The first 2 bits indicate the $''Type''$ of the flit: "00" for configuration and "11" for the spike. The

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access



(a) (b)

FIGURE 8: Network Interface: (a) Encoder encodes presynaptic spike arrays that will be transmitted from source SNPC to destination SNPCs into flits. (b) The Decoder decodes flits that arrive at a destination SNPC into spike.

next 9 bits (3 bits each for X, Y, and Z dimensions) are used to represent the address of the source neuron. The following 6 bits are a record of the time in which the source neuron fired the spike. The last 64 bits are used for the spike array from presynaptic neurons.

To ensure efficient operation, we adopt and explore the k-means based multicast routing algorithm (KMCR) and the shortest path k-means based multicast routing algorithm (SP-KMCR) presented in [40]. The KMCR routes spike packets by first dividing the destination nodes into subgroups, and within these subgroups, centroid nodes that have the least mean distance from the other nodes in the subset are selected. The packets are then routed to these centroid nodes and from these centroid nodes, using a spanning subtree, to the destination nodes. The SP-KMCR also operates in a similar manner, dividing the destination nodes into subgroups and then calculating the source's distance to all the nodes in the subgroups. However, unlike the KMCR, which uses centroids, the SP-KMCR selects a node (SP node) in the subset with a minimal distance from the source node and sends packets to it. The rest of the routing is then done from the SP node to the destination nodes.

The efficiency of such 3D mesh topology given a randomly connected (RNDC) SNN and multicast algorithm can be determined by the distance from source to destination, the efficient bandwidth, the average spike rate (SR) and the maximal spiking frequency described in equations 1-6 which are expressed in [40] as:

The total number of functional link given as:

$$TL = 3(1-\alpha)\sqrt[3]{n^2}(\sqrt[3]{n}-1). \tag{1}$$

Number of hops for each packet from source to destination:

$$TotalDist^{RNDC}_{3DMesh,MC} = C + \overline{Dist}^{RNDC} = C + 3\lambda \tag{2}$$

The efficient bandwidth:

$$BW^{RNDC}_{eff,MC} \cong \frac{\overline{w}TL}{\sqrt{n}+3\sqrt[4]{n}}.f_{NoC}.U_{NoC} \tag{3}$$

The average spike rate for each SNPC:

$$f^{MC}_{p,out} = \frac{BW^{RNDC}_{eff,MC}}{n} = \frac{3\overline{w}(1-\alpha)(\sqrt[3]{n}-1)}{\sqrt[3]{n}(\sqrt{n}+3\sqrt[4]{n})}.f_{NoC}.U_{NoC}. \tag{4}$$

The maximal spiking frequency:

$$f^{MC}_{spike,max} = \frac{s}{T_{cycle}.\overline{Dist}^{RNDC}} = \frac{s.f_{NoC}}{3\lambda} \tag{5}$$

The K ratio derived from (4) and (5):

$$K = \frac{f^{MC}_{p,out}}{f^{MC}_{spike,max}} = \frac{3\overline{w}(1-\alpha)(\sqrt[3]{n}-1)3\sqrt[4]{n}}{s.\sqrt[3]{n}(\sqrt{n}+3\sqrt[4]{n})}.U_{NoC} \tag{6}$$

Where $\overline{Dist}$ is the mean distance between two nodes, $\overline{w}$ the number of wires contained in a link, f_{NoC} the link frequency, U_{NoC} the link utilization factor, T_{cycle} the delay in the link, s the number of neurons in one SNPC, $C \cong \sqrt{n}$ is the number of connections per neuron, $\lambda \cong \sqrt[4]{n}$ is a spatial connectivity constant, and α is the fault rate in the links.

To avoid violation of the spike timing rule due to faulty links in the network, we proposed in our previous work [10] a fault-tolerant k-means based multicast routing algorithm (FT-KMCR) and a fault-tolerant shortest path k-means based multicast routing algorithm (FTSP-KMCR). To handle faulty links that lead to the violation of spike timing rules, the FT-KMCR and FTSP-KMCR as described in Fig. 9 provide alternative backup branches used to bypass faulty primary links when routing spikes in the network.

IV. EVALUATION RESULTS

The proposed system was designed in Verilog-HDL, synthesis and layout were made with Cadence tools. For ASIC implementation, we use NANGATE 45nm open-cell library [42] as the standard cells, OpenRAM [43] for generating the system memory and TSV from FreePDK3D45 [44].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

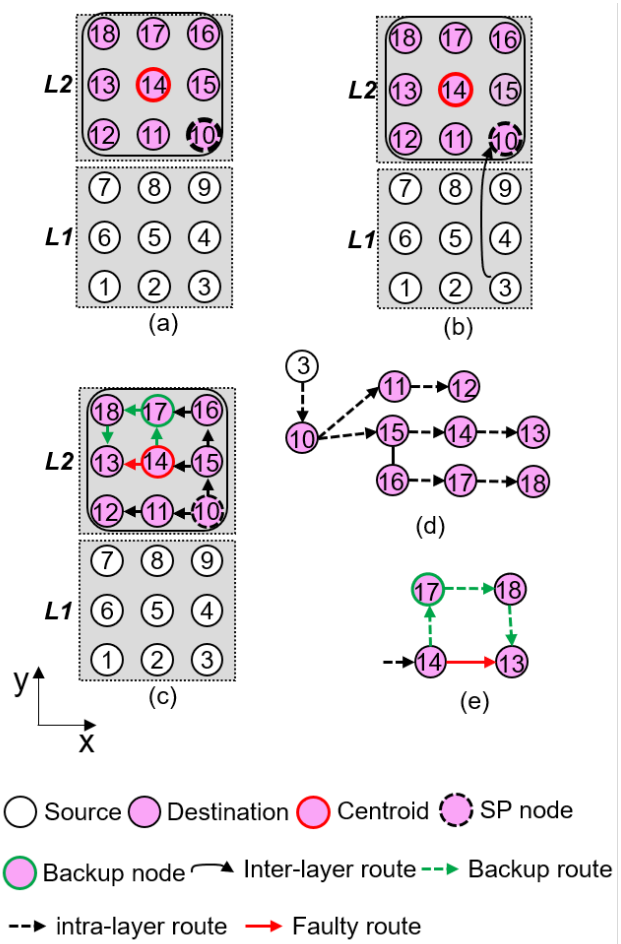


FIGURE 9: Fault-tolerant shortest path k-means based multicast routing algorithm (FTSP-KMCR) [40]. (a) Destination sub group is created and centroids identified. (b) Packets are routed from source to SP nodes. (c) From the SP nodes, packets are routed to destinations. (d) A spanning sub tree of paths from SP nodes to destinations.(e) A backup route from node 14 to node 13 through node 17 and 18, bypassing a faulty link from node 14 to 13.

To explore the efficiency of NASH, we evaluate the neurons interconnect performance using two benchmarks named Inverted Pendulum (network size of $2 \times 2 \times 3$), and Wisconsin (network size of $3 \times 3 \times 3$) [45], performing two experiments. The first experiment was carried out without faults, using the SP-KMCR, KMCR, and the baseline XYZ unicast based (UB) multicast routing algorithms. The second was carried out with faults using the FTSP-KMCR and the FT-KMCR algorithms. We also perform system performance evaluation, by classifying MNIST data set [46] with on-chip and off-chip learning. Furthermore, we evaluate NASH's hardware complexity in terms of area and power and present a comparison with some known existing works.

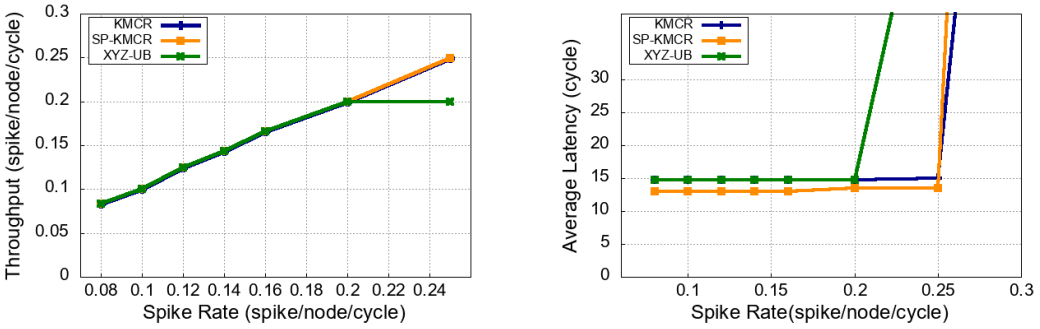
A. NETWORK PERFORMANCE EVALUATION

This subsection evaluates the network performance in terms of average latency and throughput for the SP-KMCR, KMCR, and XYZ-UB algorithms. For the first experiment, the SP-KMCR, KMCR, and the XYZ-UB as described in Fig. 10a show similar performance in terms of throughput on the Inverted Pendulum benchmark within the 0.08 and 0.2 range of SR. This is because all three algorithms can comfortably service spike traffics within this range. However, the throughput of the XYZ-UB saturates at SR of 0.2 while that of the KMCR and SP-KMCR continues to increase. As shown in Fig. 10b, their average latency slightly differs with the SP-KMCR performing better at about 10% when compared with the KMCR and XYZ-UB algorithms until SR of 0.2, thanks to the less number of hops required to reach the destinations on the SP-KMCR. At SR of 0.2, however, the XYZ-UB reaches its limit and cannot service spikes beyond this SR while the KMCR and SP-KMCR continue till 0.25 before reaching their limit. Similar to the Inverted Pendulum benchmark, the SP-KMCR, KMCR, and XYZ-UB algorithms on the Wisconsin benchmark, also show in Fig. 11a, a comparable performance in terms of average latency until SR of 0.09. Nonetheless, with the increase in network size, the SP-KMCR and KMCR in Fig. 11b show about 20% and 10% lower average latency respectively, when compared to the XYZ-UB. At SR 0.09, the XYZ-UB reaches its limit, and this can be attributed to the fact that the XYZ-UB has to repeatedly send copies of a single spike flit to all destinations, which results in high average latency. At the same time, the KMCR and SP-KMCR continue till SR of 0.11 before reaching their limit.

For the second experiment, we simulate the benchmarks with various fault rates, then measure and compare the average latency and throughput for the FT-KMCR and FTSP-KMCR algorithms. For the Inverted Pendulum benchmark, the FT-KMCR and FTSP-KMCR maintained similar performance in terms of throughput for up to 30% fault rate as shown in Fig. 12a. For the Wisconsin benchmark in Fig. 13a, throughput gradually decreases for both algorithms as the fault rate increases. The Inverted Pendulum's performance is achieved by using the first layer's intra-layer links as potential backup links, and due to the small number of nodes, there is less contention for the backup routes. However, for the Wisconsin benchmark with more nodes, more spikes are serviced, thus a higher level of contention, which decreases the throughput as the fault rate increases.

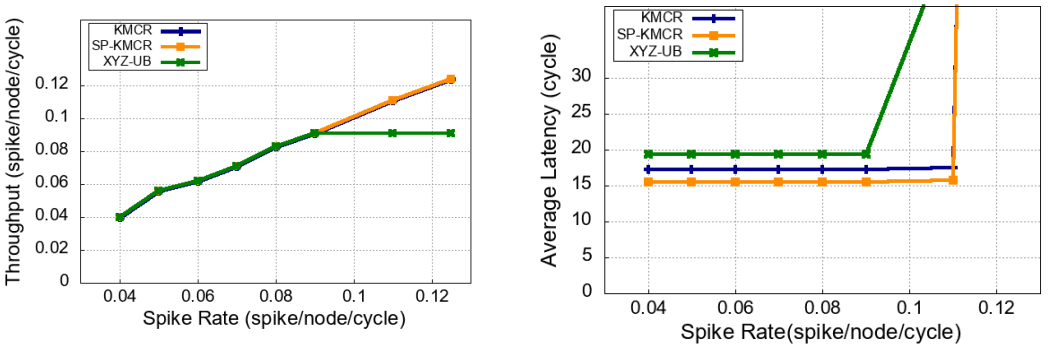
As described in Fig. 12b, the average latency on both the FT-KMCR and FTSP-KMCR for the Inverted Pendulum benchmark increases by 20 % as the fault rate reaches 30%. The same can also be observed for the Wisconsin benchmark in Fig. 13b. This increase in average latency as the fault rate increases can also be attributed to the rise in the number of spikes serviced through the backup route. The multicast-based approach easily outperform the unicast-based with much higher saturation points. Furthermore, the fault-tolerant version allows the interconnect to work under high fault rate

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access



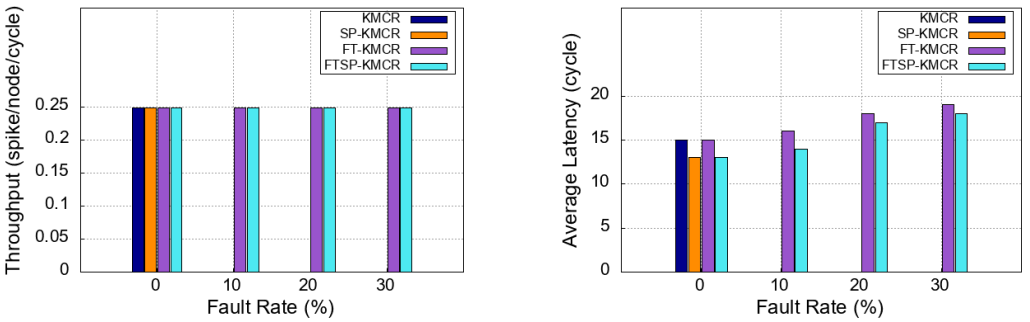
(a) (b)

FIGURE 10: Network evaluation on Inverted pendulum benchmarks over various spiking rates. (a) Throughput. (b) Latency.



(a) (b)

FIGURE 11: Network evaluation on Wisconsin benchmark over various spiking rates. (a) Throughput. (b) Latency.



(a) (b)

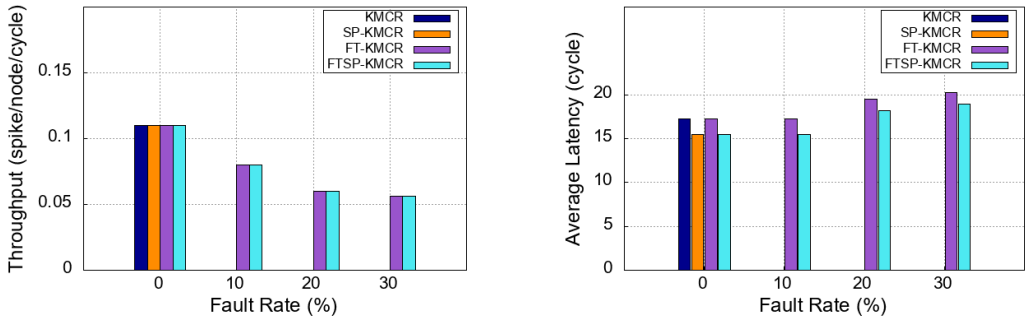
FIGURE 12: Network evaluation on Inverted pendulum benchmarks over various fault rates. (a) Throughput. (b) Average latency.

(up to 30%).

B. SYSTEM PERFORMANCE EVALUATION

To carry out the system performance evaluation, we perform MNIST dataset classification on NASH and the baseline system using SNN size of 784:225:10 that was trained offline. The MNIST benchmark was chosen for this evaluation because of its wide use, providing a basis for comparison with existing works. The MNIST images were converted to spikes with Poisson distribution using rate coding. The classification was done on NASH using a network size of $3 \times 3 \times 3$ with a layer-based mapping scheme described in Fig. 14. The input layer of 784 neurons is mapped to the first layer of NASH, utilizing 88 neurons from each of the 9 nodes in the layer. The hidden layer of 225 neurons is also mapped onto the second layer of NASH and utilized 25 neurons from each node in the layer. Finally, the output layer of 10 neurons is mapped to the third layer and utilized five neurons each from two of the nodes in the layer. For the baseline system (9×3), the input layer was mapped to the first nine vertical cores

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access



(a) (b)

FIGURE 13: Network evaluation on Wisconsin benchmark over various fault rates. (a) Throughput. (b) Average latency.

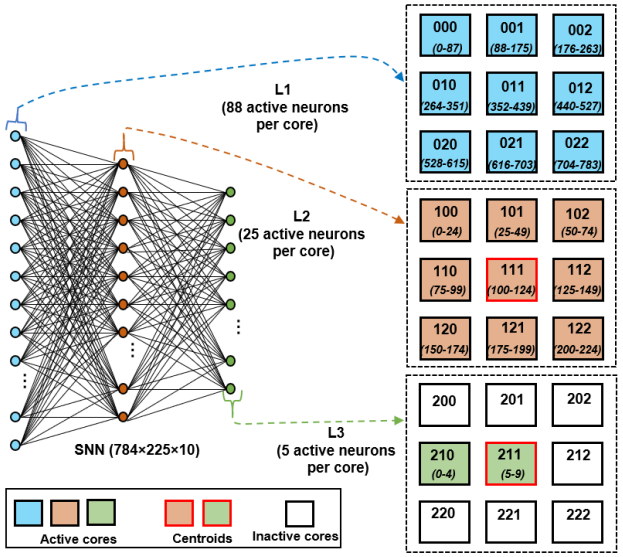


FIGURE 14: SNN mapping for MNIST classification on $3 \times 3 \times 3$ NASH: The first layer of 784 neurons without neural computation is mapped to L1 on 9 cores (88 neurons each for cores 000-021, and 80 neurons on core 022). The second layer of 225 neurons is mapped to L2 on 9 cores (25 neurons each for cores 100-122). The third layer is mapped to L3 on 2 cores (5 neurons each for cores 210 and 211).

utilizing 88 neurons from each core, the second layer to the second 9 vertical cores utilizing 25 neurons from each core, and the third layer on 2 of the third nine vertical cores using five neurons from each core.

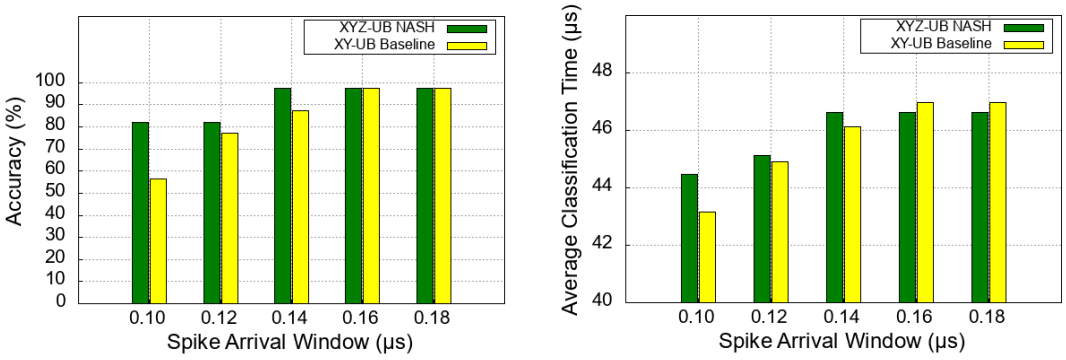
In carrying out the evaluation, we perform two experiments. The first experiment evaluates the classification accuracy and average classification time (ACT) on both NASH and the baseline system using the SP-KMCR, the XYZ-UB and the XY-UB algorithms without faults, over various spike arrival windows (SAWs). The ACT is the average time taken to classify one MNIST image, and the SAW is the time duration given for all flits (spikes) from source SNPCs to arrive at a destination SNPC. The countdown of SAW begins after the first flit arrives, and any flit that arrives after it reaches zero is not decoded. After all flits that arrived within the SAW have been decoded, the SAW is reset. For the second experiment, we evaluate the accuracy and ACT on both NASH and the baseline system using the FTSP-KMCR algorithm over various fault rates.

In the first experiment, the accuracy over various SAWs are presented for the XYZ-UB in Fig. 15a and the SP-KMCR in Fig. 16a. For the XYZ-UB, NASH at SAWs 0.1, 0.12, and 0.14, show 25.2%, 5.04%, and 10.3% better accuracy respectively, than the baseline system. Also, for the SP-KMCR, NASH at SAWs 0.1, 0.12, and 0.14, show 25.6%, 20.5%, and 10.2% better accuracy respectively, than the baseline system. This difference in accuracy is because more spikes arrived before the end of the SAWs on NASH, enabling more spikes to be processed, which resulted in better accuracy. At SAW 0.16, the accuracy on both algorithms reach 97.6% and saturates. This is because, at this SAW, all spikes for both systems arrive, and further increasing the SAW as seen at SAW 0.18 does not cause any change in the accuracy.

For the XYZ-UB in Fig. 15b, the ACT of the baseline system at SAWs 0.1, 0.12, and 0.13 despite having lower accuracy, are 2.9%, 0.6%, and 1.2% respectively lower than that of NASH, and for the SP-KMCR in Fig. 16b, the ACT of the baseline system at SAWs 0.1 and 0.12 with also lower accuracy, are 3.2% and 2.3% respectively lower than that of NASH. This is attributed to the time taken by the destination SNPC on NASH to process the increased number of spikes that arrived. However, for the XYZ-UB, at the accuracy saturation point of SAW 1.6 when all spikes arrive for NASH and the baseline system, NASH shows 1.1% lower ACT than the baseline system. For the SP-KMCR, even though NASH had reached an accuracy saturation point at SAW 0.14 and had to process more spikes than the baseline system that had not, it still shows 0.4% lower ACT. At SAW 1.6 when both NASH and the baseline system had reached saturation, NASH shows an even lower ACT of 2.5% than the baseline system.

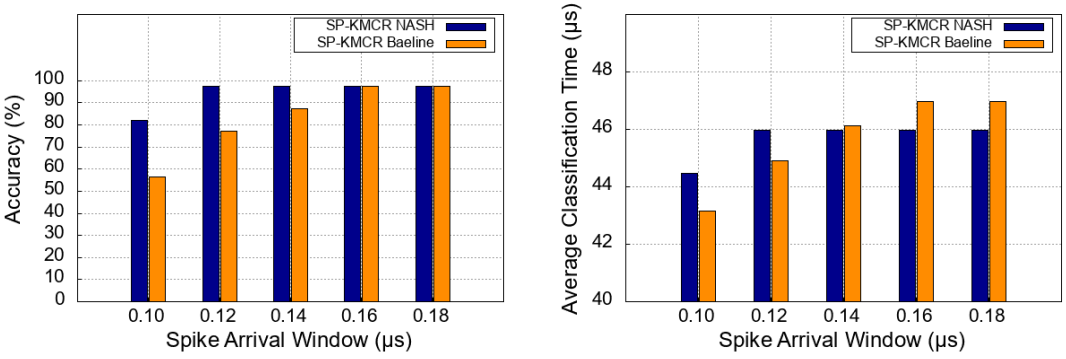
For the second experiment, the accuracy and ACT of NASH and the baseline system over various fault rates using the FTSP-KMCR algorithm are evaluated. A SAW of 0.2

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access



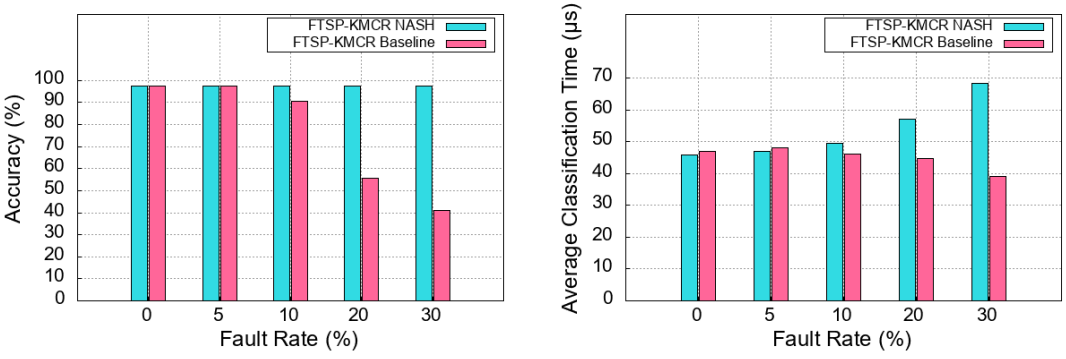
(a) (b)

FIGURE 15: MNIST classification (with off-chip learning) on NASH and the baseline system over various SAW using the XYZ-UB and XY-UB algorithms. (a) Accuracy. (b) Average classification time.



(a) (b)

FIGURE 16: MNIST classification (with off-chip learning) on NASH and the baseline system over various SAW using the SP-KMCR algorithm. (a) Accuracy. (b) Average classification time.



(a) (b)

FIGURE 17: MNIST classification (with off-chip learning) on NASH and the baseline system over various fault rate using the FTSP-KMCR algorithm. (a) Accuracy. (b) Average classification time.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

is chosen for this experiment because at this SAW, both NASH and the baseline system have reached accuracy saturation, giving enough time for the changes in the ACT to be monitored when fault rate is varied. As described in Fig. 17a, NASH and the baseline system both maintain the saturation accuracy of 97.6% from 0 to 5% fault rate. However, at 10% fault, the accuracy of the baseline system drops to 90%, and further reduces to 55.77% and 41.05% as the fault rate reaches 20% and 30% respectively. NASH on the other hand, maintains the saturation accuracy all through. The differences in accuracy between NASH and the baseline system is because a NASH node has higher path diversity than the baseline system, so in the event of faulty links, NASH has more links that can be utilized as backup compared to the baseline system, therefore delivering more spikes to the destination within the SAW than the baseline system.

In Fig. 17b the ACT for NASH and the baseline system starts at 45.96 and 46.98 microseconds respectively with zero fault rate, and slightly increases by 2% as fault rate reaches 5%. However, as the fault rate further increases, the ACT of NASH increases, while that of the baseline system decreases. At 30% fault rate, NASH utilizes 40% more ACT compared to the baseline system. The increase in ACT is due to the time taken to process the increased number of spikes that arrived within the SAW, which led to better accuracy in Fig. 17a. While the decrease in ACT for the baseline is because fewer spikes arrive within the SAW, reducing the number of spikes processed by the its destination SNPCs, which led to lower accuracy, as shown in Fig. 17a.

In conclusion, this evaluation shows the classification of MNIST dataset in our NASH system. We illustrate that with off-line training, NASH successfully delivered a reasonable accuracy. Moreover, with the help of multicast and fault-tolerance mechanisms, NASH operates with smaller SAW and faulty situations, outperforming the baseline system.

C. HARDWARE COMPLEXITY ANALYSIS

In this subsection, we evaluate and analyze NASH node's hardware complexity, and then compare with the baseline node using the XY-UB, XYZ-UB SPKMCR, and FTSP-KMCR algorithms. As described in Table 1 and Fig. 20, an FTSP-KMCR NASH node occupies a silicon area of 1.325mm^2 excluding pads. As described in Fig. 18 the synapse crossbar and SRAM-based synapse memory occupy a significant portion of the chip area at 90.3%. This is because of the considerable amount of stored synapses. All 256 neurons occupy 1.3% of the node area, the STDP learning module occupy 7.8%, and the network interface and router occupy 0.6%. At 1.1V, 25°C and a clock frequency of 142MHz, an FTSP-KMCR NASH node consumes 70.10mW. Because of the substantial amount of memory access required for moving synapse values during learning and classification, most of the energy consumed by NASH can be attributed to memory access. A comparison of NASH and the baseline system of XY-UB, XYZ-UB, SP-KMCR, and FTSP-KMCR algorithms described in Table 1 show that the NASH node of these algorithms occupy larger footprint and has higher power consumption when compared to the baseline system nodes. This is due to NASH's increased design complexity and its higher degree of path diversity enabled by TSVs, whose diameter also add to the footprint. Figure 19 evaluates how changes in synapse precision affect both the hardware complexity and the performance of NASH. With increased precision, the area, power and performance increases, and reduces otherwise.

The energy per synaptic operation (SOP) of NASH is a division of the power consumption of the SNPC and the rate of synaptic operation. NASH performs a maximum of 256 operations in one cycle, except for the STDP synapse update, which takes two cycles.

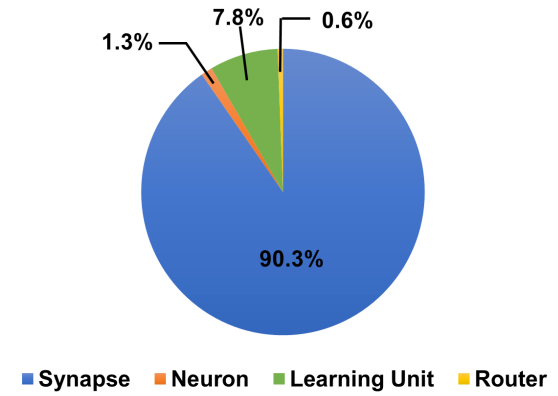


FIGURE 18: Area analysis of NASH node

D. COMPARISON WITH EXISTING WORKS

In this section, we compare the evaluation result of NASH with existing works [47]–[49]. Table 2 shows the comparison between our system and other systems. Compared to other systems, NASH is the only one that incorporates 3D-NoC fault-tolerant spike communication. Also, we support parallel LIF neurons, which allow us to perform 256 neurons' functions at the same time.

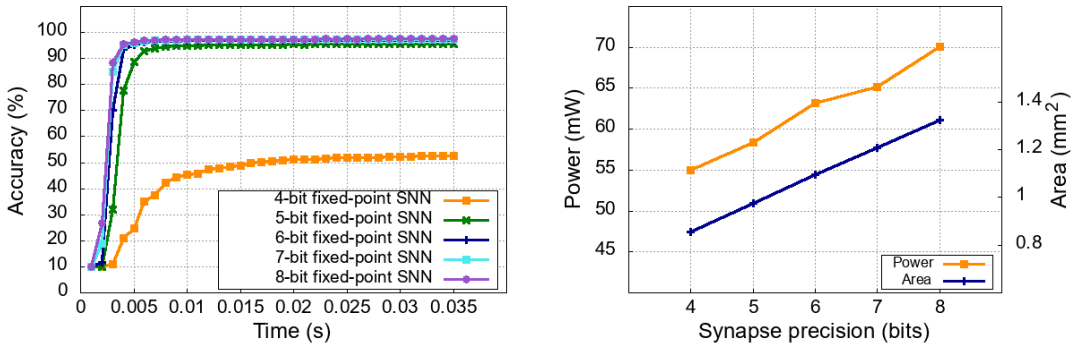
In terms of accuracy, the STDP learning method achieved an accuracy of 79.4%, which is higher than *Seo et al.* [48]. This is due to the fact we use 8-bit instead of 1-bit in the synapse SRAM. On the other hand, *ODIN* [49] and *Kim et al* [50] achieved higher accuracy. However, they both perform some pre-processing on the data set, which makes classification easier. *ODIN* in particular, employed deskewing and downsampling, which further simplified the data set. A teacher signal was also used to enhance the training.

NASH was scaled to 28nm (0.55V) with a well known scaling equation adopted from [51], and its synaptic operation (SOP) when compared with that of *ODIN* [16] as shown in Table 2, consumes less energy at 1.397pJ per SOP.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

TABLE 1: Hardware Complexity Comparison of NASH and the Baseline Nodes.

System	**XY-UB**	**XYZ-UB**	**SP-KMCR**		**FTSP-KMCR**	
Architecture	Baseline	NASH	Baseline	NASH	Baseline	NASH
Area (mm^2)	1.312	1.316	1.316	1.322	1.320	1.325
Power (mW)	66.16	66.63	66.50	66.84	68.22	70.10



(a) (b)

FIGURE 19: Performance evaluation over various synapse precision: (a) Accuracy over various synapse precision. (b) Area and power of a NASH node over various synapse precision.

TABLE 2: Comparison results between the proposed NASH and existing works.

Parameters/Systems	**Kim et al. [50]**	**ODIN [49]**	**Seo et al [48]**	**This work**
Benchmark	MNIST	MNIST	MNIST	MNIST
Accuracy (%)	84	84.5	77.2	79.4
Number of Cores	4	1	1	27
Number of Neurons / core	64	256	256	256
Neuron Model	IF	LIF and Izhikevicz	LIF	LIF
Neuron Update	serial	serial	serial	parallel
Number of Synapses /core	21k	64K	64k	65k
Synaptic Connection	2 Layer grid and ring	Crossbar	Crossbar	Crossbar
Synapse Precision	4, 5, 14	4-bit	1-bit	8-bit
Learning Rule	Stochastic gradient descent	On-chip Stochastic SDSP	on-chip STDP	On-chip STDP
Memory Technology	SRAM	SRAM	SRAM	SRAM
Interconnect	2D	2D	2D	3D-NoC
Fault-tolerant spike communication	No	No	No	Yes
Implementation	Digital	Digital	Digital	Digital
Technology	65-nm	28-nm FD-SOI CMOS	45-nm SOI-CMOS	45-nm NANGATE CMOS
Energy per synaptic operation (SOP)*(pJ)*	N/A	8.4pJ(0.55V)	N/A	11.3pJ (1.1V)
Energy per SOP at 28nm (0.55V) *(pJ)*	N/A	8.4pJ	N/A	1.397pJ

V. CONCLUSION

This paper presented the architecture, hardware design, and evaluation of a fault-tolerant 3D-NoC-based neuromorphic system with on-chip learning, named NASH. The proposed system leverages the high scalability and parallelism, low communication cost, and high throughput available in 3D-NoC to present a neuromorphic system capable of supporting large SNN with massive number of synapses. The 256 physical neurons of the SNPC as opposed to the single time-multiplexed single neuron approach enable parallel update of all neurons in a single time step. To handle challenges that may arise in spike communication and lead to performance degradation, we employed the FTSP-KMCR routing algorithm. We presented the network and performance evaluation of NASH with and without faults. From the evaluation results, we found that NASH achieved an accuracy of 79.4% and 97.6% on MNIST data set classification with on-chip and off-chip learning, respectively. Moreover, the experiments show that the proposed NASH achieves better accuracy with less ACT when compared to the baseline system and sustains a 97.6% accuracy with up to 30% fault rate while experiencing a 40% increase in the ACT as opposed to the baseline system.

As future work, we intend to further explore large-scale biological applications that will leverage NASH's potentials and reveal other aspects of its efficiency. Furthermore, these applications could provide guidelines for other functionalities that may need to be included in NASH for efficiency, fault-tolerance, and performance.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3071089, IEEE Access

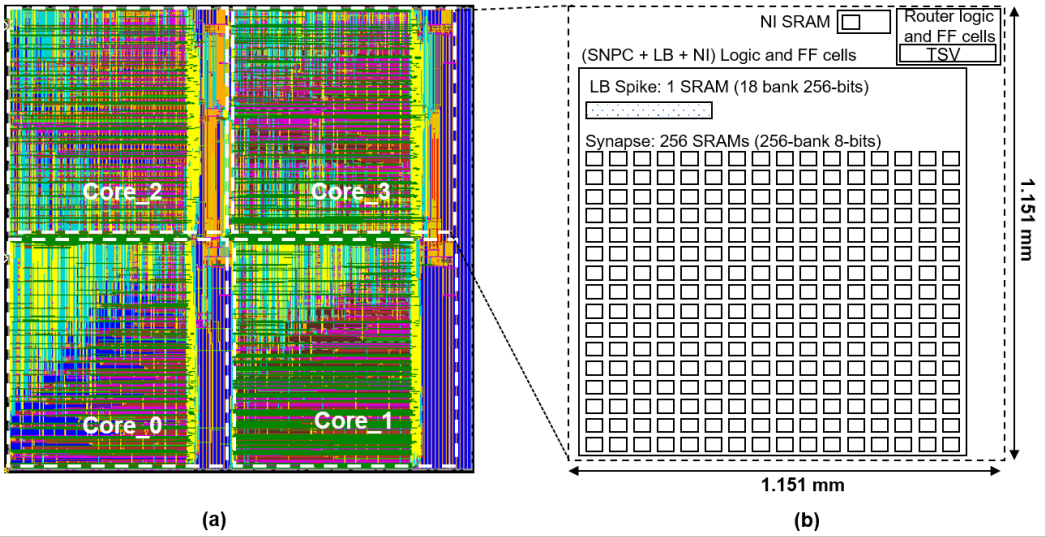


FIGURE 20: (a) Layout of a 2 × 2 NASH layer. (b) A NASH node comprising of 256 neuron logic and 65K synapses in 256 SRAMs (256-bank 8-bit), a learning module (LM) logic and spike trace SRAM (18-bank 256-bit), network interface logic and memory, and an FTMC-3DR logic and TSVs.

VI. ACKNOWLEDGEMENT

This project is supported by the University of Aizu, Competitive Research Funding (CRF), Ref. UoA-P6-2020. This work is also partially supported by the VLSI Design and Education Center, the Univ. of Tokyo, Japan, in Collaboration with Synopsys, Inc., and Cadence Design Systems, Inc.

REFERENCES

[1] M. Hopkins, G. García et al., "Spiking neural networks for computer vision," Interface Focus, vol. 8, no. 4, pp. 128–136, Jun. 2018.

[2] D. Valencia, J. Thies, and A. Alimohammad, "Frameworks for efficient brain-computer interfacing," IEEE Transactions on Biomedical Circuits and Systems, vol. 13, no. 6, pp. 1714–1722, Dec. 2019.

[3] Z. Bing, C. Meschede et al., "A survey of robotics control based on learning-inspired spiking neural networks," Frontiers in Neurorobotics, vol. 12, pp. 436–457, 2018.

[4] B. Sen-Bhattacharya, S. James et al., "Building a spiking neural network model of the basal ganglia on SpiNNaker," IEEE Transactions on Cognitive and Developmental Systems, vol. 10, no. 3, pp. 823–836, Sep. 2018.

[5] W. Maass, "Networks of spiking neurons: The third generation of neural network models," Neural Networks, vol. 10, no. 9, pp. 1659–1671, Dec. 1997.

[6] J. Rodrigues de Oliveira Neto, J. P. Cerquinho Cajueiro, and J. Ranhel, "Neural encoding and spike generation for spiking neural networks implemented in FPGA," in 2015 International Conference on Electronics, Communications and Computers (CONIELECOMP), 2015, pp. 55–61.

[7] N. Fourcaud-Trocmé, Encyclopedia of Computational Neuroscience: Integrate and Fire Models, Deterministic. New York, NY: Springer New York, 2013, pp. 1–9.

[8] M. F. Bear, B. W. Connors, and M. A. Paradiso, Neuroscience: Exploring the Brain, 4th Edition. Lippincott Williams and Wilkins, 351 W Camden St, Baltimore, MD 21201, United States: Lippincott Williams and Wilkins, 2016, pp. 81–108.

[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] 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.

[11] M. Henry, "The blue brain project," Nature Reviews Neuroscience, vol. 2, no. 7, pp. 153–159, 2006.

[12] S. B. Furber, D. R. Lester et al., "Overview of the spinnaker system architecture," IEEE Transactions on Computers, vol. 62, no. 12, pp. 2454–2467, Dec. 2013.

[13] K. N. Dang, A. B. Ahmed et al., "Scalable design methodology and online algorithm for TSV-cluster defects recovery in highly reliable 3D-NoC systems," IEEE Transactions on Emerging Topics in Computing, vol. 8, no. 3, pp. 577–590, Oct. 2017.

[14] S. Furber and S. Temple, "Neural systems engineering," Journal of The Royal Society Interface, vol. 4, no. 13, pp. 193–206, nov 2006.

[15] S. Carrillo, J. Harkin et al., "Scalable hierarchical network-on-chip architecture for spiking neural network hardware implementations," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 12, pp. 2451–2461, Dec. 2013.

[16] C. Frenkel, J.-D. 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, vol. 13, pp. 999–1010, Oct. 2019.

[17] M. Davies, N. Srinivasa et al., "Loihi: A neurophic manycore processor with on-chip learning," IEEE Micro, vol. 38, no. 1, pp. 82–99, January 2018.

[18] N. Qiao, H. Mostafa et al., "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses," Frontiers in Neuroscience, vol. 9, pp. 39–55, 2015.

[19] F. Akopyan, J. Sawada et al., "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, 2015.

[20] S. Yang, J. Wang et al., "Real-time neuromorphic system for large-scale conductance-based spiking neural networks," IEEE Transactions on Cybernetics, vol. 49, no. 7, pp. 2490–2503, 2019.

[21] B. Belhadj, A. Valentian et al., "The improbable but highly appropriate marriage of 3D stacking and neuromorphic accelerators," in 2014 International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES), Oct. 2014, pp. 1–9.

[22] S. Yang, B. Deng et al., "Scalable digital neuromorphic architecture for large-scale biophysically meaningful neural network with multi-compartment neurons," IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 1, pp. 148–162, 2020.

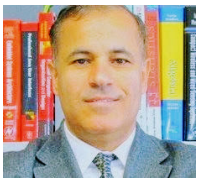
[23] D. S. Bassett and E. T. Bullmore, "Small-world brain networks revisited," The Neuroscientist, vol. 23, no. 5, pp. 499–516, Sep. 2016.

[24] N. Ohno, M. Katoh et al., "Recent advancement in the challenges to connectomics," Microscopy (Oxford, England), vol. 65, pp. 97–107, Dec. 2015.

[25] D. S. Bassett and E. Bullmore, "Small-world brain networks," The Neuroscientist, vol. 12, no. 6, pp. 512–523, Dec. 2006.
[26] S. Shibata, Y. Komaki et al., "Connectomics: Comprehensive approaches for whole-brain mapping," Microscopy, vol. 64, pp. 57–67, Feb. 2014.
[27] T. Leergaard, C. Hilgetag, and O. Sporns, "Mapping the connectome: Multi-level analysis of brain connectivity," Frontiers in Neuroinformatics, vol. 6, pp. 6–11, 2012.
[28] P. Wijesinghe, A. Ankit et al., "An all-memristor deep spiking neural computing system: A step toward realizing the low-power stochastic brain," pp. 345–358, Oct. 2018.
[29] Q. Xia and J. J. Yang, "Memristive crossbar arrays for brain-inspired computing," Nature Materials, vol. 18, no. 4, pp. 309–323, Mar. 2019.
[30] A. Balaji, A. Das et al., "Mapping spiking neural networks to neuromorphic hardware," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 1, pp. 76–86, 2020.
[31] A. Taherkhani, A. Belatreche et al., "A supervised learning algorithm for learning precise timing of multiple spikes in multilayer spiking neural networks," IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 11, pp. 5394–5407, 2018.
[32] A. R. Young, M. E. Dean et al., "A review of spiking neuromorphic hardware communication systems," IEEE Access, vol. 7, pp. 135 606–135 620, 2019.
[33] A. Ben Abdallah, 3D Integration Technology for Multicore Systems On-Chip. Singapore: Springer Singapore, Sep. 2017, pp. 175–199.
[34] C. Torres-Huitzil and B. Girau, "Fault and error tolerance in neural networks: A review," IEEE Access, vol. 5, pp. 17 322–17 341, 2017.
[35] brainchip, "Akida NSoC," https://brainchipinc.com/automotive/, 2021, (accessed 25.02.2021).
[36] M. A. Ehsan, Z. Zhou, and Y. Yi, "Modeling and analysis of neuronal membrane electrical activities in 3D neuromorphic computing system," in 2017 IEEE International Symposium on Electromagnetic Compatibility Signal/Power Integrity (EMCSI), Aug 2017, pp. 745–750.
[37] K. N. Dang, A. B. Ahmed et al., "TSV-OCT: A scalable online multiple-TSV defects localization for real-time 3-D-IC systems," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 3, pp. 672–685, 2019.
[38] M. A. Ehsan, Hongyu An et al., "Adaptation of enhanced TSV capacitance as membrane property in 3D brain-inspired computing system," in 2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC), Jun. 2017, pp. 1–6.
[39] G. Indiveri, B. Linares-Barranco et al., "Neuromorphic silicon neuron circuits," Frontiers in Neuroscience, vol. 5, pp. 697–720, 2011.
[40] 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," ACM Journal on Emerging Technologies in Computing Systems, vol. 15, no. 4, pp. 1–28, Dec. 2019.
[41] M. Rahimi Azghadi, N. Iannella et al., "Spike-based synaptic plasticity in silicon: Design, implementation, application, and challenges," Proceedings of the IEEE, vol. 102, no. 5, pp. 717–737, May 2014.
[42] NanGate Inc., "Nangate Open Cell Library 45 nm," http://www.nangate.com/, (accessed 23.02.2021).
[43] M. R. Guthaus, J. E. Stine et al., "Openram: An open-source memory compiler," in 2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), vol. 34, no. 2, 2017, pp. 1–6.
[44] NCSU Electronic Design Automation, "FreePDK3D45 3D-IC process design kit," http://www.eda.ncsu.edu/wiki/FreePDK3D45:Contents, (accessed 23.02.2021).
[45] S. Cawley, F. Morgan et al., "Hardware spiking neural network prototyping and application," Genetic Programming and Evolvable Machines, vol. 12, pp. 257–280, Sep. 2011.
[46] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database," http://yann.lecun.com/exdb/mnist/, (accessed 23.02.2021).
[47] Y. Kim, Y. Zhang, and P. Li, "A reconfigurable digital neuromorphic processor with memristive synaptic crossbar for cognitive computing," ACM Journal on Emerging Technologies in Computing Systems, vol. 11, no. 4, pp. 1–25, Apr. 2015.
[48] J. Seo, B. Brezzo et al., "A 45nm cmos neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," in 2011 IEEE Custom Integrated Circuits Conference (CICC), Sep. 2011, pp. 1–4.
[49] C. Frenkel, M. Lefebvre et al., "A 0.086-mm^2 12.7-pj/SOP 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm CMOS," IEEE Transactions on Biomedical Circuits and Systems, vol. 13, no. 1, pp. 145–158, Feb 2019.
[50] J. K. Kim, P. Knag et al., "A 640m pixel/s 3.65mw sparse event-driven neuromorphic object recognition processor with on-chip learning," in 2015 Symposium on VLSI Circuits (VLSI Circuits), 2015, pp. C50–C51.
[51] A. Stillmaker and B. Baas, "Scaling equations for the accurate prediction of CMOS device performance from 180 nm to 7 nm," Integration, the VLSI Journal, vol. 58, pp. 74–81, 2017.

OGBODO M. IKECHUKWU is currently a Ph.D. student at the Adaptive Systems Laboratory (ASL), the University of Aizu, Japan. He received his master's degree in computer science from the African University of Science and Technology (AUST) Abuja, Nigeria in 2017, and his BSc. in computer science from Ebonyi State University Abakaliki, Nigeria in 2015. His current research interests are in the field of Neuromorphic and Machine Learning Systems.

KHANH N. DANG is currently an assistant professor at VNU Key Laboratory for Smart Integrated Systems, Vietnam National University Hanoi (VNU), Hanoi Vietnam. He received his B.Sc., M.Sc., and Ph.D. degree from VNU University of Engineering and Technology, University of Paris-Sud XI, and The University of Aizu, Japan in 2011, 2014, and 2017, respectively. His research interests include System-on-Chips/Network-on-Chips, 3D-ICs, neuromorphic computing, and fault-tolerant systems. Dr. Khanh N. Dang was visiting researcher at University of Aizu in 2019 and 2020-2021.



ABDERAZEK BEN ABDALLAH (Senior Member, IEEE) Abderazek Ben Abdallah is a Professor in the graduate and undergraduate schools of computer science and engineering at the University of Aizu, Aizu-Wakamatsu, Japan. He is concurrently Head of the Division of Computer Engineering at the school of computer science and engineering. He has a Ph.D. degree in computer engineering from the University of Electro-Communications at Tokyo in 2002, an MS degree in computer engineering, and a BS degree in electrical engineering from Huazhong University of Science and Technology in 1994 and 1997, respectively. From 2002 to 2007, Dr. Ben Abdallah was a faculty member at the University of Electro-communications. Since October 2007, he has been with the School of Computer Science and Engineering at the University of Aizu. Dr. Ben Abdallah conducts research in the area of computer systems and parallel computing, with an emphasis on adaptive/self-organizing systems, on/off-chip interconnection networks, and reliable multicore systems-on-chip. His current research interest lies in studying neural processing systems with a particular focus on spike-based neural network dynamics and spike-based learning. He is a senior member of ACM.