Scaled-up Neuromorphic Array Communications Controller (SNACC) for Large-scale Neural Networks

Aaron R. Young, Adam Z. Foshie,
Mark E. Dean, James S. Plank, Garrett S. Rose
Department of Electrical Engineering and Computer Science
University of Tennessee
Knoxville, Tennessee, USA, 37996
Email: [ayoung48, afoshie]@vols.utk.edu
[markdean, jplank, garose]@utk.edu

J. Parker Mitchell, Catherine D. Schuman
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831-6085
Email: [mitchelljp1, schumancd]@ornl.gov

***Abstract*—Neuromorphic computing is one promising post-Moore's law era technology, which takes inspiration from biological brains to perform computing tasks. The human brain contains billions of neurons with trillions of synapses and as neuromorphic hardware systems scale to larger and larger sizes, the communication system used to transfer information between neuromorphic elements and traditional computers must scale to keep up. In prior work, we describe the use of a separate neuromorphic array communications controller to support low-latency, high-throughput communication between our neuromorphic systems and a traditional computer. In this work, the neuromorphic array communications controller is used to support the scaling of a neuromorphic development system which uses multiple neuromorphic processors arranged in a two-dimensional array. The neuromorphic array communications controller, along with scalable local connections, is used to create a scalable neuromorphic platform to enable the development and testing of large neuromorphic network arrays.**

I. Introduction

As the limits of semiconductor physics and Moore's law are reached, new architectures that break away from the traditional von Neumann architecture and traditional circuit design will have to be researched and developed to continue to push the frontier of computing [1]. There are many fundamental limitations to the potential scaling of traditional computing, including the physics of increasingly smaller circuitry, the increasing heat generated by these systems, and the von Neumann bottleneck found in the separation of data storage and computation [1], [2]. One promising class of post-von Neumann architectures are brain-inspired, neuromorphic architectures, such as Spiking Neural Networks (SNN)s. SNNs rely on event-based, time-dependent, binary spikes to transmit information. These neural networks form a complex graph where the neurons are the nodes of the graph, and the synapses are the edges. There are many different neuron types, but they typically accumulate charge received from their synapses and decide when to fire based on the accumulated charge compared to a threshold value. When the neuron fires a binary pulse is sent to the output synapses. The synapses transmit the binary signal from the pre-synaptic neuron to the post-synaptic neuron and apply a weight value that represents the strength of the connection. These weight values are then accumulated by the post-synaptic neuron when a fire event occurs.

Spikes, neurons, and synapses are all modeled after the electrical pulses transmitted amongst neurons in biological brains. Many models exist; some try to be biologically realistic by modeling the behavior of the brain exactly and are used to further our understanding of the brain's operation, while others seek to build biologically inspired computers where the focus is less on building an accurate biological brain simulator but instead on using insights from the brain to build powerful computing devices in various technologies. The latter approach acknowledges that the brain has evolved to be a very efficient biological computer and insights from its organization can be used to build powerful computing systems without modeling all of the biological systems found therein. There is still disagreement among researchers on the level of detail required in the models to successfully emulate the workings of the brain [3]. Neuromorphic systems avoid the von Neumann bottleneck by colocating computational elements with memory. Furthermore, they avoid the heat issue caused by the

Notice: This material is based in-part on research sponsored by The University of Tennessee (UT) Science Alliance Joint Directed Research and Development Program; Air Force Research Laboratory under agreement number FA8750-19-1-0025; and by the U.S. Department of Energy, Office of Science, and Office of Advanced Scientific Computing Research, under contract number DE-AC05-00OR22725. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government. This manuscript has been authored in part by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

end of Dennard scaling by using small distributed processors spread across the chip. Additionally, novel technologies provide alternative design options beyond CMOS [1], [4]–[6].

The human brain is made up of approximately 86 billion neurons, each with 1,000 to 10,000 synaptic connections to other neurons resulting in trillions of synapses [7]. The brain is clever because of the massive connectivity between its many neurons [8]. One major design challenge is how to build a neuromorphic communication system which can scale to support the massive number of connections found in biology.

Many SNNs are implemented as separate neuromorphic processors, either within a Field-Programmable Gate Array (FPGA) [9], as a custom Very Large-Scale Integration (VLSI) chip [10], or through the use of an emerging technology [6]. In order to scale these processors, a method to connect them together and to a traditional von Neumann based computer is necessary. This connection must be reliable and fast in order to support deterministic operation and to operate at the max performance of the neuromorphic processors. The connection between the neuromorphic processors must support the transfer of time-dependent spiking information. The connection to the traditional computer must support real-time operations to enable the neuromorphic system to be used as a coprocessor able to run real-time data processing and control tasks.

We propose a solution to the scaling communication challenge with a globally asynchronous, locally synchronous (GALS) approach [11]. The neuromorphic processors employed are synchronous and deterministic. The communication system is designed to asynchronously transfer information between the devices while maintaining the high performance, deterministic operation of the neuromorphic processors. The processors working together are built into a large neuromorphic array. The sub-array communication system uses a series of high-performance, point-to-point connections in the cardinal directions to build up a two-dimensional communication mesh. Communication between the neuromorphic array and a traditional von Neumann machine, known as the host system, is established through the use of a neuromorphic array communications controller (NACC) to build a hierarchical communication tree that allows the host to interface with the neuromorphic processors making up the array.

This paper discusses the considerations and design of the Scaled-up Neuromorphic Array Communications Controller (SNACC) built to support scalable neuromorphic arrays using a multitude of individual neuromorphic processors operating asynchronously from one another. Scaling up neuromorphic hardware in this way is tremendously important for the development and testing of large neuromorphic systems and can be used as a step toward developing larger custom VLSI designs. The design decisions and accompanying details are intended to be useful for those who want to develop a communications system to support the parallel operation of multiple neuromorphic processors. This system is the first of its kind to support the scaling-up of multiple neuromorphic processors to construct large neural networks while maintaining deterministic, MHz range operation, with an accompanying network-cycle-accurate simulator. The performance evaluation of the scaled-up neuromorphic hardware systems and the software simulator is measured and compared in order to provide insight into the advantages and drawbacks of each approach.

II. Related Work

Various research groups are tackling the challenge of designing neuromorphic systems. The vast majority of the groups have considered how the system can scale and have worked to build large scale systems. Each of these groups have had to tackle the issue of how to transfer a large amount of spiking information generated by the neuromorphic cores. Although the characteristics of each neuromorphic design vary, they each had to solve the issue of routing packets at increasing larger scales.

Researchers at Stanford University have developed two separate neuromorphic systems. The first is Neurogrid, a mixed-signal system with millions of neurons and billions of synapses, able to perform biological real-time simulations for computational neuroscientists [12]. A full Neurogrid system consists of 16 Neurocores connected in a tree structure. Each Neurocore contains a 256×256 array of analog neurons fabricated in 180-nm CMOS. The second system is Braindrop [13]. Like Neurogrid, Braindrop uses mixed-signal neurons. However, Braindrop is designed to be programmed at a higher level of abstraction through the use of the Neural Engineering Framework. The Braindrop chip uses fractal H-trees for routing to be integrated into a larger chip called Brainstorm [14].

The Human Brain Project (HBP) in Europe has the goal of "building a research infrastructure to help advance neuroscience, medicine, and computing" [15]. Research by a collaboration of groups including the University of Heidelberg and the Technische Universität Dresden have developed BrainScaleS, a mixed-signal waferscale neuromorphic hardware system [16], [17]. BrainScaleS is built up from Analog Network Cores (ANC) combined together to create a High Input Count Analog Neural Network (HICANN). 352 of these HICANN chips are fabricated onto a $20\,\mathrm{cm}$ wafer using 180-nm CMOS. An intra-wafer mesh router is used for local communication while a hierarchical routing tree is used for inter-wafer communication with the help of digital network chips (DNC) [16], [18]. The same group is working on BrainScaleS-2 which supports many additional features including a more complex neuron model, nonlinear dendrites, and structured neurons [19]. Researchers at the University of Manchester have built a large digital neuromorphic system called SpiNNaker [20]. SpiNNaker simulates the brain using over one million parallel ARM processors simulating neurons and synapses in real-time. Eighteen ARM processors, fabricated in 130-nm CMOS, are organized into chip multiprocessors (CMPs). 2^{16} CMPs are then networked together in a two-dimensional toroidal mesh structure. The same group is now working on a second version of SpiNNaker called SpiNNaker2 with improved ARM processors [21].

Researchers at IBM have developed TrueNorth as part of the DARPA SyNAPSE program [22]. TrueNorth consists of digital integrate-and-fire neurons arranged in cores with 256 neurons and 1,024 axonal circuits. 4,096 of these cores are implemented on a single chip. TrueNorth packets are routed to neighboring cores connected in a two-dimensional mesh network.

Loihi is a digital neuromorphic research chip recently developed by Intel [23], [24]. Each Loihi chip has 128-neuromorphic cores, as well as three Lakemont cores. Loihi's design allows for on-chip SNN learning rules to be implemented with a unique programmable microcode running in the cores and with the help of the Lakemont cores for advanced learning rules. Loihi is fabricated with Intel's 14-nm process, and each chip implements 130,000 artificial CUBA leaky-integrate-and-fire neurons and 130 million synapses. The cores communicate using a two-dimensional grid, which is extended to neighboring chips.

Research groups at Zhejiang University and Hangzhou Dianzi University, both in China, have built the small scale Darwin Neural Processing Unit (NPU), which is targeted for resource constrained embedded systems [25], [26]. Darwin's eight physical neurons, which simulate 2048 logical neurons, use an off-chip memory to route packets to the different physical neurons.

Dynap-SEL is a mixed-signal multi-core architecture of neuromorphic processors created by researchers at the University of Zurich in Switzerland, in the lab of Giacomo Indiveri [27], [28]. This architecture has a novel routing scheme which allows memory requirements to scale with the number of neurons in a way much lower than other standard routing schemes. This is accomplished through the use of a mixed tag-based shared-addressing routing scheme which combines the use of point-to-point and broadcast routing, as well as hierarchical and grid routing.

In [4], Young *et al.* explore the design of the large-scale spiking communication networks used in these seven well-known neuromorphic systems, as well as, the the considerations unique to spiking packet communication. Three broad categories of routing schemes are used to route packets between neuromorphic elements: Mesh, hierarchical tree, and memory routing. Two main routing methods are employed: source-based routing and destination based routing. Figure 1 summarizes how these various systems designed their routing methods.

This work also builds upon prior work conducted by TENNLab at the University of Tennessee. Most significantly, prior work on host co-processor communication to the Dynamic Adaptive Neural Network Array (DANNA) digital neuromorphic processor [29], [30] and work for Tiled DANNA, a project used to tile multiple DANNA processors in an array with shared global clocks [31]. Both influenced the decisions made in the development of SNACC. The DANNA2 neuromorphic architecture is also central to this work as it serves as the neuromorphic processor used within the array [32], [33].

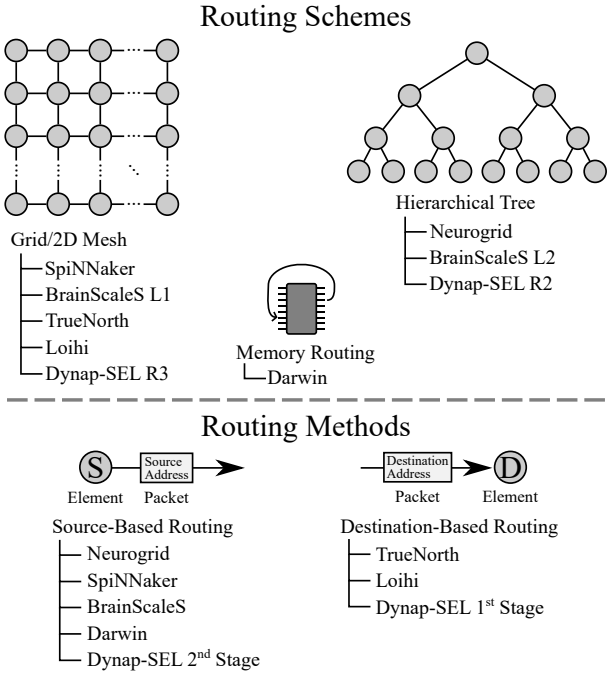

Fig. 1. Graphical summary of neuromorphic hardware communication systems. The top half of this figure summarizes the routing schemes used by the neuromorphic systems, and the bottom half summarizes the routing methods [4].

III. DANNA2

DANNA2 is the second iteration of the DANNA digital neuromorphic processor designed by TENNLab and is used to implement the neural network arrays within SNACC [32], [33]. DANNA2 is specifically designed to map well to FPGA and VLSI designs with the logic written as generic VHDL. Many design decisions of DANNA2 help drive the SNACC design. DANNA2 is made up of a two-dimensional grid of elements. Each element implements a single leaky-integrate and fire neuron with 24 synapses. These synapses are connected to the 24 nearest neighboring elements. The DANNA2 FPGA implementation operates with a very fast network update which occurs every ten MHz, much faster than other neuromorphic systems that have network cycles in the kilohertz range [34]. This implementation is provided with a single 100 MHz global clock, and the element calculations are completed every ten of these global cycles. The elements are programmable, allowing for different neural network configurations to be loaded. Each element outputs a binary signal from the neuron which is transferred to the connected element's synapses. Input to the array is provided to elements on the left side of the array and output fires are read from the right side of the array.

IV. SNACC Design

The main SNACC design objective is to create a development platform for neuromorphic hardware which allows larger neuromorphic systems to be developed and tested by combining multiple neuromorphic processors together to extend their capacity and functionality. Additionally, SNACC should place

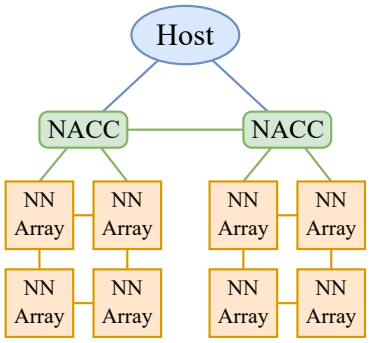

Fig. 2. SNACC Overview

minimal restrictions on the neuromorphic systems and behave as though it was a single neuromorphic array with a much greater capacity. Although SNACC was built with the DANNA2 neuromorphic core, this core can be replaced with other neuromorphic processors.

The SNACC design comprises three major components, shown in Figure 2. The first component is the neuromorphic arrays which are tiled together to form a large array. Between the neuromorphic sub-arrays are point-to-point sub-array communication channels which allow local fire information to be shared among the sub-arrays. The second is the Neuromorphic Array Communications Controller (NACC) which is used to send packets between the neuromorphic array and the host. Multiple NACCs can be used in a hierarchical tree to continue scaling the size of the neuromorphic array. Communication controller packets are sent between the NACC and the array. The last component is the host machine, which is a traditional computer used to control the operations of, provide input to, and interpret the output from the network. Host packets are sent between the host and NACC.

SNACC is designed with a GALS design pattern. The host, NACC, and neuromorphic sub-arrays are all synchronous, however, each of these systems is asynchronous to one another, i.e., each driven with an independent clock. Each communication channel is likewise clocked independently. Clock converters are used to transfer packets between clock domains. The GALS pattern allows for larger scaling than would be possible within a single clock domain and for the communication channels to run at different clock speeds than the neuromorphic core. Martin *et al.* have noted that future SoCs and large scale designs will no longer be able to operate under a single clock because the variations across a large chip or multiple large chips will make it prohibitively expensive to attempt to manage the delays in a clock and other global signals [11].

With a globally asynchronous approach, timing of events is no longer guaranteed. In order to maintain deterministic behaviour, a synchronization mechanism must be included. The synchronization system used by SNACC comes from the observation that the evaluation of the current cycle only depends on information from prior cycles. Each element progresses to the next cycle once all the cycle information upon which it is

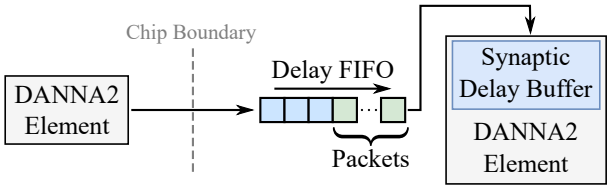

Fig. 3. Delay FIFO is used to move the Synaptic Delay from inside the element to make it part of the communications channel.

dependent has been received. This decouples wall-time from simulation time and makes the hardware operate similar to a distributed event processor. Now that wall-time and simulation time are decoupled, the time needed to evaluate a given number of cycles is no longer guaranteed. This prevents the ability to make hard real-time guarantees. However, since the system operates much faster than real-time, soft real-time guarantees can be made based on the statistical likeliness of meeting the timing requirement.

Another important consideration is the performance of the system. After each neuromorphic processor finishes a cycle, it sends its output to its neighboring processors. When a neuromorphic processor has received input from each of its neighbors, it has the required information to start evaluating the next cycle. Ideally, the neuromorphic processors will constantly be evaluating the next timestep and will not need to wait around for input from previous cycles. This will allow SNACC to run at the same speed as the individual neuromorphic processors. Although the sub-array communication channels support enough throughput to send a new packet each cycle, the latency of the packet being sent requires multiple cycles before arrival. This issue can be alleviated by enforcing a minimum synaptic delay and by using a FIFO to make the communications channel add this delay instead of the neuromorphic element's synaptic delay buffer. Figure 3 shows how a FIFO external to the synaptic delay buffer allows the sub-array communication latency to be hidden by increasing the minimum synaptic delay. To initialize the FIFO with the correct delay value, the number of starting packets in the FIFO is equal to the minimum synaptic delay plus one. The max size of the FIFO is twice the number of starting packets. This design allows the neuromorphic processors to continue operating at peak performance since the latency of the communication channel is hidden by the minimum synaptic delay. Furthermore, this design is easy to implement since it only requires injecting blank packets into the FIFO when the system is reset, and removing the minimal delay from the configuration of the synaptic delay buffer when the elements are configured.

In order to verify the logical design of SNACC, a simulator was designed to simulate the asynchronous communication patterns of the SNACC system using software. The simulator is event-based with a 1 nanosecond time resolution and simulates the communication patterns of each communication channel. This simulator demonstrated that the SNACC system is able to run at effectively peak performance by using a minimum synaptic delay to hide the sub-array performance latency.

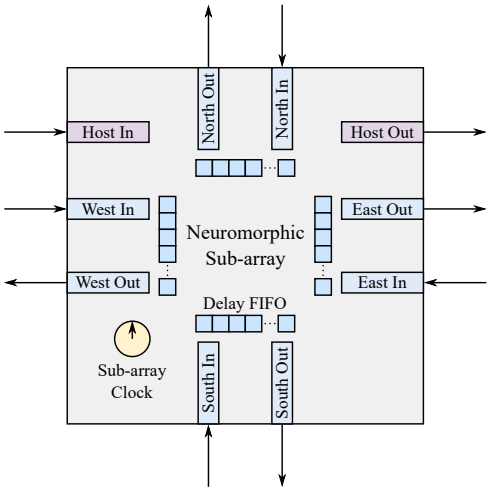

Fig. 4. Logical view of the neuromorphic sub-array.

Figure 4 shows the logical view of each sub-array component used by the simulator. Each component has connections to its neighbors and the host, a local clock to consume and generate packets, and delay FIFOs.

V. SNACC Implementation

The Aurora 64B/66B protocol from Xilinx is used to build the communication channels between the FPGAs used to implement NACC and the neuromorphic sub-arrays. The Aurora core "is a scalable, lightweight, high-data rate, link-layer protocol for high-speed serial communication" [35]. Aurora uses the high-speed transceivers available on the FPGAs to transmit and receive data. Since Aurora is only a link level protocol, it does not have any provision for guaranteeing in-order, error-free transmission of packets. Therefore, a custom version of the Go-Back-N retransmission protocol was implemented. This custom design is crafted to work well with both Aurora and AXI4-Stream. AXI4-Stream is the streaming handshake bus protocol used throughout SNACC within the FPGAs. AXI4-Stream is one of the protocols defined by the Advanced eXtensible Interface 4 (AXI4) protocols which are part of the ARM Advanced Microcontroller Bus Architecture 4 (AMBA4).

The sub-array components are designed such that each sub-array can use the same design. Switches on the FPGA board are used to direct packets to the appropriate sub-array communication channel based on where the board is located in the array. The sub-arrays are designed with five bidirectional Aurora channels. One channel is used to connect to the NACC and the other four channels are used to connect to neighboring boards to the north, south, east, and west. The design also includes retransmission logic and packet FIFOs. The FIFOs are used to buffer incoming and outgoing packets as well as to convert between the communication channel's clock domain and the neuromorphic processor's clock domain. The FIFO and clock domain crossing is implemented with custom logic to reduce the latency added to the sub-array communication's critical path. The channels for sub-array communication also have logic to initialize the FIFOs with an initial number of packets to implement the channel delay. DANNA2 neuromorphic processors were adjusted to receive and send fire packets to neighboring DANNA2 processors. These changes were made so that minimal connectivity constraints were placed on the system. The only additional constraint added is that diagonal connections at the corners of the sub-arrays are not allowed. In order to allow these connections the number of sub-array connections or the sub-array latency would have to be doubled. The sub-array communication was the most challenging component to design for SNACC. It was validated and tuned with test benches and debug probes to achieve peak sub-array communication performance.

The NACC component is implemented to allow the host to communicate via PCIe to each of the neuromorphic sub-arrays. Xillybus is used to transfer packets from the host to the FPGA using the Xillybus Linux driver and hardware design [36]. NACC then sends the packets to the correct sub-array based on the Xillybus stream. More complex routing methods can be added to NACC as the number of sub-arrays is increased. NACC connects to the sub-arrays using Aurora with the custom retransmission logic. Large FIFOs on NACC allow the PCIe to send large bursts of data at a time.

The host system must have an updated Xillybus driver to communicate with the NACC over PCIe. The TENNLab software framework was also extended with libraries to include SNACC as one of the supported neuromorphic devices [37].

VI. Results

As SNACC was developed, each component was verified, both individually and as part of the system. Once the individual communication components were all verified, the entire SNACC system was built and tested. SNACC was first implemented with a false neuromorphic core which behaved like the logical view of the core used by the simulator. The false core would keep track of its current time step, send and receive full-sized packets, and would generate and accept packets at the correct rate. This implementation was used to verify that the behavior of SNACC matched the expected behavior from the simulator.

The effect that varying the number of added delay cycles had on the average element clock cycle frequency was both modeled using the SNACC simulator and measured on the hardware. The results are shown in Figure 5. As expected from the simulator and from hand calculations, five added delay cycles are required to hide the array-to-array communication latency and to allow a multi-board DANNA2 array to operate with the same performance as a single-board DANNA2 array.

Once the communication systems were verified, the entire system's operation was verified. SNACC performed as expected with the DANNA2 cores, and deterministic operation was achieved with the SNACC output matching the DANNA2 simulator's output for the same network and input pattern. The SNACC hardware system is shown in Figure 6. Currently, the SNACC system is built with a two-by-two board array; however, this size was chosen to be a proof of concept and larger arrays can be built. The logical design of the system is such that any

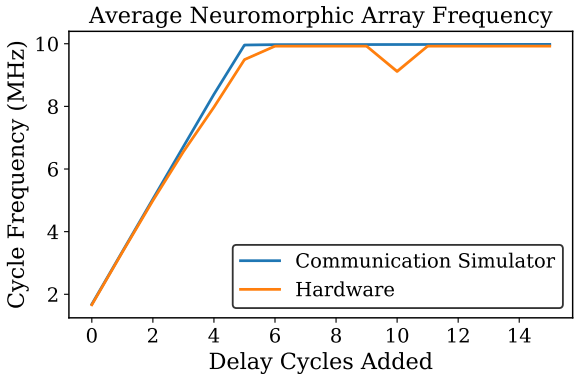

Fig. 5. Added delay verses average element clock cycle frequency.

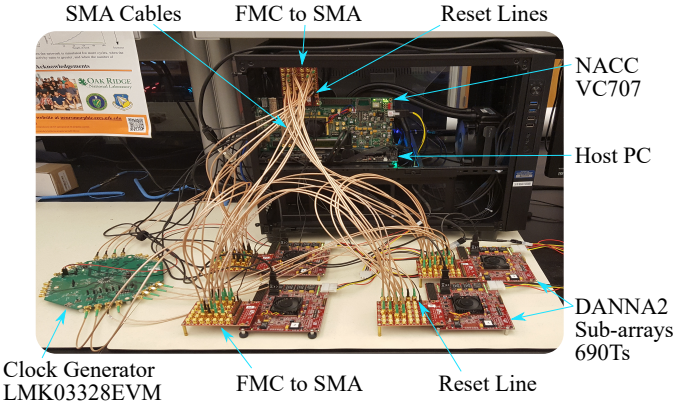

Fig. 6. SNACC Hardware Setup

arbitrary number of sub-arrays can be supported. The system was built using a computer running Ubuntu 16.04, powered by an AMD Threadripper 1950X, ASUS ROG Zenith Extreme EATX motherboard, and 32 GB of DDR4-3600 memory. NACC was implemented on a Xilinx Virtex-7 FPGA VC707 evaluation kit (VC707). Each sub-array was implemented on a HiTech Global HTG-777 Stackable Development Platform with a Xilinx Virtex-7 X690T FPGA (690T). The sub-arrays were connected together via SMA cables using HiTech Global's 8-port SMA-FMC modules. NACC is connected to the host via a PCIe port.

In order to determine the performance characteristics of the DANNA2 simulator versus the DANNA2 hardware, a series of test networks of various sizes were evaluated and timed. Figure 7 shows these graphs. In each graph, the elapsed real-world time in seconds is shown on the vertical axis. The horizontal axes show the height of the neural network. The width of the network was increased as well, to be roughly half of the height. Each network was filled with elements in a passthrough or snake pattern.

With the passthrough pattern, every element in the first column is an input neuron and receives fire information; likewise, every element in the last column generates output. The tests were conducted on three systems: the DANNA2

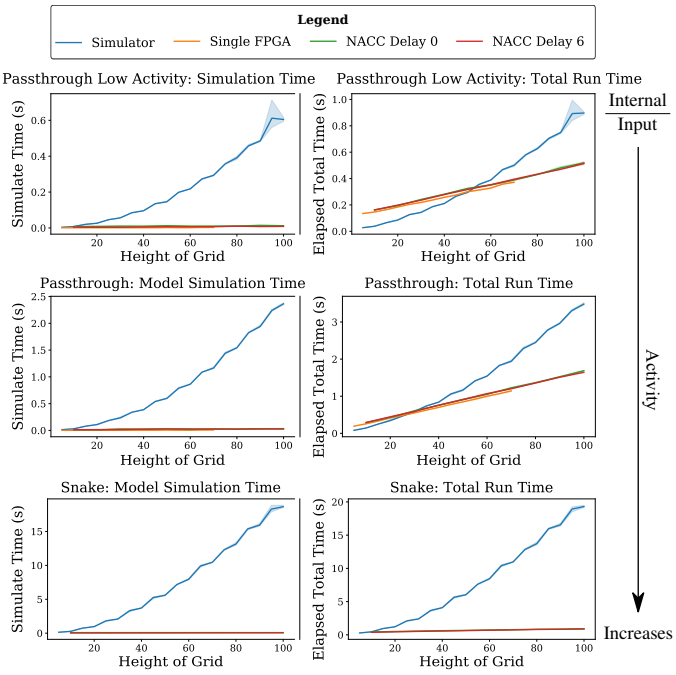

Fig. 7. Network evaluation time comparison between hardware and software

simulator, DANNA2 on a single FPGA, and DANNA2 on a 2-by-2 SNACC system. The SNACC system was tested both with no added synaptic delay and with an added synaptic delay of six. The graphs on the top row show the elapsed wall-clock time for running a passthrough network with low activity. A passthrough network connects each element to the one on its right, such that the input is received from the left side and is passed through the network to outputs on the right side. The graphs in the first column focus only on the time required for the network evaluation portion of the run. The graphs in the second column show the total time for the entire run, which includes the time required to generate input packets for the hardware test points.

As seen in these graphs, the time required to evaluate the array in hardware did not change, since each network was run for the same total amount of network cycles in each case. In the case of the total runtime, the time required for the hardware changed linearly with the number of input elements due to the additional input packets required by them. The software simulator increased at a rate linearly correlated with the total number of elements in the array. This is expected since the software simulator is an event simulator, and increasing the number of elements would lead to an increase in the number of events to process. The width of the array is the closest multiple of five which is half of the height; this explains the wavy increases in the simulation evaluation time. Going down the rows of the graph, the relative amount of internal activity over input activity increases. Passthrough low activity received an input fire every 4 cycles. The passthrough run in the middle row receives an input fire every cycle. This causes the simulator to outperform the hardware with smaller network sizes. (Lower

elapsed time is better.)

The use of a snake network in the final row results in the best relative performance of hardware. The snake network only has one input, but every element in the array is used. Since only one input is used for all the sizes, the total simulation time is roughly linear. The snake network is close to the ideal network to run on hardware in terms of hardware outperforming software. This is in contrast to sparse networks with few fires performing best on the software simulator, where fewer events need to be simulated.

VII. Simulator Versus Hardware Discussion

The DANNA2 simulator and the hardware evaluate the neuromorphic networks differently. The simulator uses an event queue and processes events as they happen. This has the side effect of allowing the simulator to run incredibly fast when there are few events. The time required to evaluate a given timestep depends on the number of events that occur at that timestep. This means the simulator is particularly suited for evaluating large, sparse networks with relatively few fires. The simulator treats input fires the same way as internal fires, so there is no additional penalty for an event being an input or an internal fire.

The hardware simulator, on the other hand, evaluates each timestep at a fixed frequency. Since the array is implemented in hardware, the events of each element are evaluated in parallel, resulting in a fixed timestep evaluation time. Array input is not the same as internal fires; internal fires are generated and consumed with the hardware elements, whereas input fires originate from the host, and the host must convert each of the fire events into a fire packet to be sent to the hardware. This extra processing results in the number of input fires having a larger impact on the total running time of the system. Therefore, the hardware is best suited for running jobs which utilize dense arrays with a large amount of internal activity compared to the amount of input activity.

VIII. Future Work

The SNACC system has demonstrated that high-performing neuromorphic processors operating with a ten MHz network clock can be built into a large, multi-board, neuromorphic system with local sub-array communication and hierarchical connection back to a host system with the use of a separate communications controller. The next steps are to demonstrate further scaling by using one NACC to communicate with a larger number of sub-arrays. After the capabilities of a single NACC board are exhausted, the system can be further extended with multiple NACCs organized in a hierarchy to support more sub-array boards. Another direction would be to explore replacing the DANNA2 cores with more advanced neuromorphic cores. Further work is also being conducted using SNACC to evaluate the benefits of running larger than previously possible DANNA2 networks in hardware.

IX. Conclusion

A new communications system for neuromorphic network arrays, which tiles together multiple neuromorphic processors, was designed, verified, and evaluated. The Scaled-up Neuromorphic Array Communications Controller is able to support large-scale neural networks through the use of multiple point-to-point connections used to combine individual neuromorphic processors together, with each one implementing a piece of the large neural network. Each of these neural sub-arrays are then connected, via point-to-point streaming connections, back to the host through a neuromorphic communications controller. This system is the first of its kind to link together independently clocked neuromorphic processors running with a ten MHz network cycle using high-speed, point-to-point connections, all while guaranteeing error-free deterministic operation of the resulting scaled-up network.

References

[1] M. M. Waldrop, "The chips are down for moore's law," *Nature*, vol. 530, no. 7589, pp. 144–147, Feb. 2016. DOI: 10.1038/530144a.

[2] J. Backus, "Can programming be liberated from the von neumann style?: A functional style and its algebra of programs," *Commun. ACM*, vol. 21, no. 8, pp. 613–641, Aug. 1978, ISSN: 0001-0782. DOI: 10.1145/359576.359579. [Online]. Available: http://doi.acm.org/10.1145/359576.359579.

[3] H. Markram, "The blue brain project," *Nature Reviews Neuroscience*, vol. 7, no. 2, p. 153, 2006.

[4] A. R. Young, M. E. Dean, J. S. Plank, and G. S. Rose, "A review of spiking neuromorphic hardware communication systems," *IEEE Access*, vol. 7, pp. 135 606–135 620, 2019. DOI: 10.1109/ACCESS.2019.2941772.

[5] M. S. Hasan, C. D. Schuman, J. S. Najem, R. Weiss, N. D. Skuda, A. Belianinov, *et al.*, "Biomimetic, soft-material synapse for neuromorphic computing: From device to network," in *IEEE 13th Dallas Circuits and Systems Conference (DCAS)*, Nov. 2018. DOI: 10.1109/DCAS.2018.8620187. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8620187.

[6] S. Buckley, A. N. McCaughan, J. Chiles, R. P. Mirin, S. W. Nam, J. M. Shainline, *et al.*, "Design of superconducting optoelectronic networks for neuromorphic computing," in *IEEE International Conference on Rebooting Computing*, Tysons, VA, Nov. 2018, pp. 36–42.

[7] F. A. C. Azevedo, L. R. B. Carvalho, L. T. Grinberg, J. M. Farfel, R. E. L. Ferretti, R. E. P. Leite, *et al.*, "Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain," *Journal of Comparative Neurology*, vol. 513, no. 5, pp. 532–541, 2009.

[8] J. Wu, S. Furber, and J. Garside, "A programmable adaptive router for a gals parallel system," in *2009 15th IEEE Symposium on Asynchronous Circuits and Systems*, May 2009, pp. 23–31. DOI: 10.1109/ASYNC.2009.17.

[9] M. E. Dean, J. Chan, C. Daffron, A. Disney, J. Reynolds, G. S. Rose, *et al.*, "An application development platform for neuromorphic computing," in *International Joint Conference on Neural Networks*, Vancouver, Jul. 2016.

[10] M. E. Dean and C. Daffron, "A VLSI design for neuromorphic computing," in *IEEE Annual Symposium on VLSI (ISVLSI)*, IEEE, Jul. 2016. DOI: 10.1109/ISVLSI.2016.81.

[11] A. J. Martin and M. Nystrom, "Asynchronous techniques for system-on-chip design," *Proceedings of the IEEE*, vol. 94, no. 6, pp. 1089–1120, Jun. 2006, ISSN: 0018-9219. DOI: 10.1109/JPROC.2006.875789.

[12] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J.-M. Bussat, *et al.*, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, 2014.

[13] A. Neckar, S. Fok, B. V. Benjamin, T. C. Stewart, N. N. Oza, A. R. Voelker, *et al.*, "Braindrop: A mixed-signal neuromorphic architecture with a dynamical systems-based programming model," *Proceedings of the IEEE*, vol. 107, no. 1, pp. 144–164, Jan. 2019, ISSN: 0018-9219. DOI: 10.1109/JPROC.2018.2881432.

[14] S. Fok and K. Boahen, "A serial h-tree router for two-dimensional arrays," in *2018 24th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, May 2018, pp. 78–85. DOI: 10.1109/ASYNC.2018.00026.

[15] Human Brain Project. (Apr. 10, 2019). Short overview of the human brain project, [Online]. Available: https://www.humanbrainproject.eu/en/about/overview/.

[16] J. Schemmel, D. Briiderle, A. Griibl, M. Hock, K. Meier, and S. Millner, "A wafer-scale neuromorphic hardware system for large-scale neural modeling," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, May 2010, pp. 1947–1950. DOI: 10.1109/ISCAS.2010.5536970.

[17] S. Scholze, S. Schiefer, J. Partzsch, S. Hartmann, C. Mayr, S. Höppner, *et al.*, "Vlsi implementation of a 2.8 gevent/s packet-based aer interface with routing and event sorting functionality," *Frontiers in Neuroscience*, vol. 5, p. 117, 2011, ISSN: 1662-453X. DOI: 10.3389/fnins.2011.00117. [Online]. Available: http://journal.frontiersin.org/article/10.3389/fnins.2011.00117.

[18] S. Scholze, H. Eisenreich, S. Höppner, G. Ellguth, S. Henker, M. Ander, *et al.*, "A 32gbit/s communication soc for a waferscale neuromorphic system," *Integration, the VLSI Journal*, vol. 45, no. 1, pp. 61–75, 2012.

[19] J. Schemmel, "Towards the second generation brainscales system," in *NICE Workshop*, 2018.

[20] S. Furber and A. Brown, "Biologically-inspired massively-parallel architectures-computing beyond a million processors," in *Ninth International Conference On Application of Concurrency to System Design, 2009. ACSD'09.*, IEEE, Jul. 2009, pp. 3–12. DOI: 10.1109/ACSD.2009.17. [Online]. Available: https://eprints.soton.ac.uk/270985/1/PID871138.pdf.

[21] S. Hoppner, "Spinnaker2, Towards extremely efficient digital neuromorphics and multi-scale brain emulation," in *Nice Workshop*, 2018.

[22] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, *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, Oct. 2015, ISSN: 0278-0070. DOI: 10.1109/TCAD.2015.2474396.

[23] M. Davies, N. Srinivasa, T. Lin, G. Chinya, Y. Cao, S. H. Choday, *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, Jan. 2018, ISSN: 0272-1732. DOI: 10.1109/MM.2018.112130359.

[24] Wikichip. (Aug. 8, 2019). Loihi - intel, [Online]. Available: https://en.wikichip.org/wiki/intel/loihi.

[25] J. Shen, D. Ma, Z. Gu, M. Zhang, X. Zhu, X. Xu, *et al.*, "Darwin: A neuromorphic hardware co-processor based on spiking neural networks," *Science China Information Sciences*, vol. 59, no. 2, pp. 1–5, Feb. 2016, ISSN: 1869-1919. DOI: 10.1007/s11432-015-5511-7. [Online]. Available: https://doi.org/10.1007/s11432-015-5511-7.

[26] D. Ma, J. Shen, Z. Gu, M. Zhang, X. Zhu, X. Xu, *et al.*, "Darwin: A neuromorphic hardware co-processor based on spiking neural networks," *Journal of Systems Architecture*, vol. 77, pp. 43–51, 2017, ISSN: 1383-7621. DOI: https://doi.org/10.1016/j.sysarc.2017.01.003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1383762117300231.

[27] S. Moradi, N. Qiao, F. Stefanini, and G. Indiveri, "A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (dynaps)," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 12, no. 1, pp. 106–122, Feb. 2018, ISSN: 1932-4545. DOI: 10.1109/TBCAS.2017.2759700.

[28] C. S. Thakur, J. L. Molin, G. Cauwenberghs, G. Indiveri, K. Kumar, N. Qiao, *et al.*, "Large-scale neuromorphic spiking array processors: A quest to mimic the brain," *Frontiers in Neuroscience*, vol. 12, p. 891, 2018, ISSN: 1662-453X. DOI: 10.3389/fnins.2018.00891. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2018.00891.

[29] A. R. Young, "Scalable high-speed communications for neuromorphic systems," Master's thesis, University of Tennessee, 2017.

[30] A. R. Young, M. E. Dean, J. S. Plank, G. S. Rose, and C. D. Schuman, "Neuromorphic array communications controller to support large-scale neural networks," in *IJCNN: The International Joint Conference on Neural Networks*, Rio de Janeiro, Brazil, Jul. 2018.

[31] P. J. Eckhart, "Tiled DANNA: Dynamic adaptive neural network array scaled across multiple chips," Master's thesis, University of Tennessee, 2017.

[32] J. P. Mitchell, M. E. Dean, G. Bruer, J. S. Plank, and G. S. Rose, "DANNA 2: Dynamic adaptive neural network arrays," in *International Conference on Neuromorphic Computing Systems*, Knoxville, TN: ACM, Jul. 2018. DOI: 10.1145/3229884.3229894.

[33] J. P. Mitchell, "Danna2: Dynamic adaptive neural network arrays," Master's thesis, University of Tennessee, Aug. 2018.

[34] 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. DOI: 10.1109/CICC.2011.6055294.

[35] Xilinx, *Aurora 64b/66b v12.0, Logicore ip product guide*, PG046, version 11.0, Xilinx, May 22, 2019. [Online]. Available: https://www.xilinx.com/support/documentation/ip_documentation/aurora_8b10b/v11_0/pg046-aurora-8b10b.pdf (visited on 06/14/2017).

[36] Xillybus Ltd., *Ip core product brief*, Feb. 16, 2017. [Online]. Available: http://xillybus.com/downloads/xillybus_product_brief.pdf (visited on 06/12/2017).

[37] J. S. Plank, C. D. Schuman, G. Bruer, M. E. Dean, and G. S. Rose, "The TENNLab exploratory neuromorphic computing framework," *IEEE Letters of the Computer Society*, vol. 1, no. 2, pp. 17–20, Jul. 2018. DOI: 10.1109/LOCS.2018.2885976. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/LOCS.2018.2885976.