Hardware Acceleration for Neuromorphic Computing: An Evolving View

Beiye Liu, Xiaoxiao Liu, Chenchen Liu, Wei Wen, M. Meng, Hai Li, and Yiran Chen
Department of Electrical and Computer Engineering
University of Pittsburgh
Pittsburgh, USA
{bel34, xil116, chl192, wew57, mem231, hal66, yic52} @pitt.edu

***Abstract*— The rapid growth of computing capacity of modern microprocessors enables the wide adoption of machine learning and neural network models. The ever-increasing demand for performance, combining with the concern on power budget, motivated the recent research on hardware acceleration for these learning algorithms. A wide spectrum of hardware platforms have been extensively studied, from conventional heterogeneous computing systems to emerging nanoscale systems. In this paper, we will review the ongoing efforts at Evolutionary Intelligence Laboratory (www.ei-lab.org) about hardware acceleration for neuromorphic computing and ma-chine learning. Realizations on various platforms such as FPGA, on-chip heterogeneous processors, and memristor-based ASIC designs will be explored. An evolving view of the accelerator de-signs for learning algorithms will be also presented.**

Keywords-component: Neuromorphic, Memrisotr, Accelerator

I. INTRODUCTION

With increasing demand of high performance computation, the conventional Von Neumann computer architecture becomes less efficient. Recently, a lot of research attentions have been put on neuromorphic hardware systems that can efficiently provide the capabilities of biological perception and information processing [1][2]. As a highly generalized and simplified abstract of a biological system, an artificial neural network (ANN) connects two groups of neurons with synaptic connections. The weights of the connections are called “weight matrix” in this paper. Accordingly, the transformation between input and output neurons can be described as matrix-vector multiplication(s). Therefore, hardware realizations of neural networks require a large volume of memory and are associated with high design complexity and hardware cost [2]. Traditional von Neumann computers require frequent data exchanging between processors and memory chips. This design severely limits the system performance and efficiency, especially in computation-intensive cognitive applications.

Various hardware implementations of ANNs have been proposed across both digital and analog domains. Examples include neural network accelerators for signal processing [3], digital approximate computing accelerators that leverage neural network algorithms [4], and heterogeneous systems built with GPUs and CPUs for deep learning accelerations [5]. However, traditional CMOS technology has scaling limitations for neuromorphic system design, as many transistors are usually required to build one neuron [5].

Discovery of nanoscale memristor devices [12] inspired an exciting approach to implement neuromorphic systems. Particularly, the similarity between the programmable resistance state of memristors and the variable weight of biological synapses dramatically simplify the circuit realization of ANN. The memristor devices have been investigated and exploited in a few research works that focus on either the circuit implementation of the matrix-vector multiplications in conventional approximate computing acceleration[18][19][16][17]. In this work, we will review the ongoing efforts at Evolutionary Intelligence Laboratory (EI-Lab) on memristor-based neuromorphic system from various design levels, e.g., computing model, circuit design and computer architectures.

We will first introduce the Brain-state-in-a-box (BSB) computing model and use its circuit design as a demonstration. The BSB model is an auto-associative neural network that is commonly used in optical character recognition (OCR) for printed text [6]. An input character image is processed through the BSB models in parallel for the recall (pattern recognition) operation. When all recalls are completed, a set of candidates are selected based on the convergence speed. In our design, input signals can be represented as voltages and memristor crossbar will naturally transfer the weighted combination of input signals to output voltages. A group of several memristor crossbars can be used as a single neuromorphic processing unit (NPU) along with other NPUs.

In order to organize multiple memristor crossbars to work as a complete accelerator we also proposed a novel efficient reconfigurable neuromorphic computing accelerator architecture. Our accelerator uses on-chip memristor crossbar based NPUs to implement perceptron networks, aiming at the acceleration of ANN computations. Unlike many neuromorphic systems that perform the computations on pure digital ALUs or analog approximate computing units with AD/DA interface, our design adopts a hybrid method in data representation: the computation within the crossbars and the signal communications between crossbars are conducted in analog form, while the control information remains as digital signals. Compared to the existing implementations of digital ANN accelerators and approximate computing units, our accelerator speeds up neuromorphic computing and support the implementations of a variety of neural network topologies.

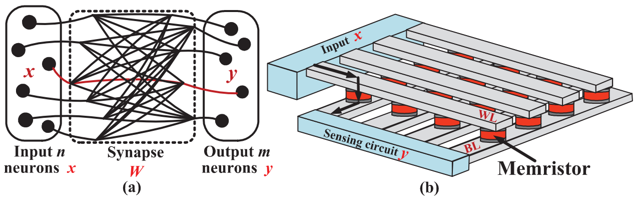

Figure. 1: (a) Artificial neural network. (b) Memristor crossbar [20].

II. PRELIMINARY

A. *Memristor Crossbar Basics*

Predicted by Prof. Leon Chua, the memristor is the fourth fundamental circuit element uniquely defining the relationship between magnetic flux and electrical charge [12]. In 2008, HP Labs reported that the memristive effect was realized by moving the doping front along a T_iO_2 thin-film device [7]. The resistance of a memristor can be controlled by adjusting the magnitude and pulse width of programmed current/voltage.

Figure1 (a) depicts a conceptual overview of a neural network that can be implemented with a memristor crossbar-based neuromorphic computing system. Two groups of neurons are connected by a set of synapses. Input neurons send voltage signals $\boldsymbol{x}$ to memristor crossbar, as shown in Figure 1 (b). The output neurons collect the information (current) from the input neurons through the synapses (memristors in a crossbar) and process them with an activation function. The synapses apply different weights (synaptic strengths) on the information during the transmission. In general, the relationship between the input neurons $\boldsymbol{x}$ and the output $\boldsymbol{y}$ can be described as [6]:

$$\boldsymbol{y} = \boldsymbol{W} \cdot \boldsymbol{x} \tag{1}$$

Here the weight matrix $\boldsymbol{W}$ denotes the weight matrix of the connections between the two neuron groups. Equation (1) is widely used in the operations of the weighted connections in neural networks.

B. *Program mechanism*

The memristors of a crossbar can be programmed by applying proper voltage pulses on the metal wires. For example, when programing the memristor $\boldsymbol{w_{ij}}$, we may apply a positive voltage v on the *i-th* horizontal wire and ground the *j-th* vertical wire. To avoid interference to other memristors, all the unselected wires are connected to $v/2$ so that only the memristor $\boldsymbol{w_{ij}}$ has a full voltage bias v while all the other memristors in the crossbar are half selected with a voltage of $v/2$ [20]. Because of the nonlinearity of the voltage dependency in memristor programming, the resistances of the half-selected memristors remain almost unchanged during the programming. Before deploying an ANN, the connection weight matrix $\boldsymbol{W}$ is pre-calculated and memristor are programmed accordingly.

III. LEVEL BASED DESIGN WITH BSB EXAMPLE

There are many ways to represent signals in neuromorphic systems. One popular way is to code the signal as strength of voltage/current. We call this representation a level-based design. In the level based design, the input signals of a MBC

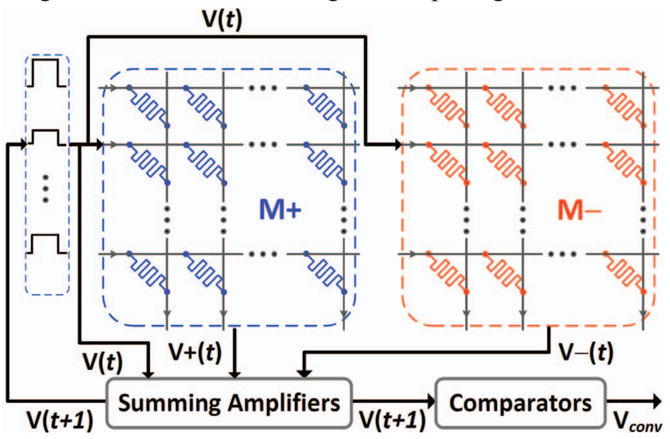

Figure 2. Memristor crossbar based implementation of BSB model [15].

can be either digital or analog, while output current through each column of MBC is always analog. Hence, the input voltage and output current has to be processed with digital-analog converter (DAC) and analog-digital converter (ADC). Here we use a Brain-State-in-a-Box (BSB) model to demonstrate our design.

The mathematic model of the recall function of the BSB model is described as:

$$\boldsymbol{x}(t+1) = S(\,\alpha \cdot \boldsymbol{W} \cdot x(t) + \lambda \cdot \boldsymbol{x}(t)\,). \tag{2}$$

Here, $\boldsymbol{W}$ is pre-calculated connection matrix that is mapped on to memristor crossbars. $\boldsymbol{x}(t)$ and $\boldsymbol{x}(t+1)$ are input/output signals, S is the sigmoid function, α and λ are learning rate parameters. In the recall process of this model, operation of Equation (2) is repeated until $\boldsymbol{x}(t+1)$ converged to a stable state that output of each column are either "1" or "0". In order to perform this loop, we build feedback circuit that send $\boldsymbol{x}(t)$ back to the input ports. The schematic of BSB hardware realization is shown in Figure 2. The main components are memristor crossbars, summing amplifier and comparators, etc... As the connection weights of $\boldsymbol{W}$ in Equation (1) can be either positive or negative, we have to use two crossbar to represent one $\boldsymbol{W}$. The weights in $\boldsymbol{W}$ are separated into positive matrix "$\boldsymbol{W}^{+}$" that only contains positive weights and negative matrix "$\boldsymbol{W}^{-}$" that only contains negative ones. Now the Equation (2) becomes:

$$V(t+1) = S\,(\,\boldsymbol{W}^{+} \cdot V(t) - \boldsymbol{W}^{-} \cdot V(t) + V(t)). \tag{3}$$

Here $V(t)$ and $V(t+1)$ are the input voltage of crossbar and the output voltage of the summing amplifier. The summing amplifier collects and sum signal from $\boldsymbol{W}^{+} \cdot V(t)$, $\boldsymbol{W}^{+} \cdot V(t)$ and $V(t)$, then amplifies them to a proper range for post process, which is a comparator here. The comparator is used here to implement the sigmoid function in the Equation (3). The "+" terminal of the comparator is set to $V(t+1)$ while the "-" terminal is target value, e.g., *Vdd* or *Gnd*. When $V(t+1)$ is equal or larger than *Vdd*, the computation converges and finishes. Otherwise, $V(t+1)$ will be used as the input of next round of computation.

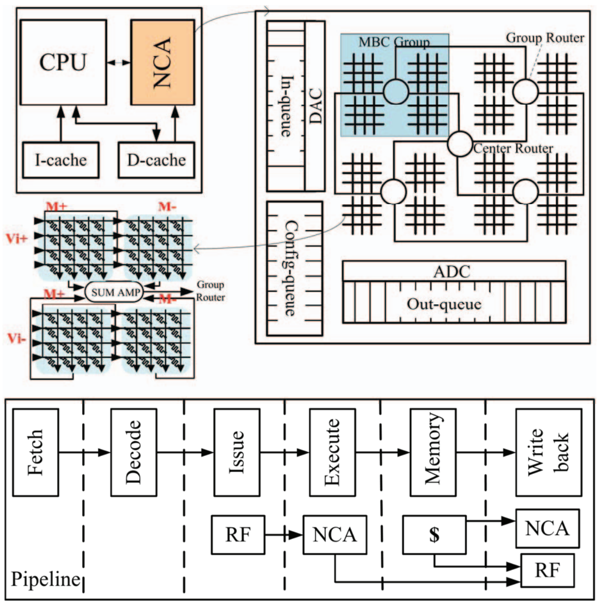

Figure 3. Neuromorphic computing accelerator [19].

IV. Neuromorphic computing accelarator

A. *Design Overview*

Our proposed neuromorphic computing accelerator (NCA) design is shown in the Figure 3. In one NCA, four memristor crossbar based computing groups are placed in a metamorphous centralized Mesh (MCMesh) manner, which are connected through a central router. Each group consists of four identical memristor crossbars, each of which has a size of 128×128 by assuming each layer of a neural network contains no more than 128 neurons. As neural networks are mostly used as classifier of extracted features, such array scale is sufficient to cover nearly 90% of learning applications [21]. Further increasing the size of single crossbar offers very little benefit on the computation performance improvement but incurs exponential circuit reliability degradation [20]. As memristor can only represent positive weights, four crossbars in the same group needs to work together to implement the multiplication between signed signals and signed synaptic weights, e.g., negative inputs are multiplied with positive weights by the MBC at the left-bottom corner.

Without loss of generality, we use a weight matrix $\boldsymbol{M_{n\times m}}$ as an example to demonstrate the mapping of the weight matrix on the crossbar arrays. In this case, *n* input neurons and m output neurons are connected together with $\boldsymbol{M_{n\times m}}$. If *max (n,m)* $\leqslant$ *128*, the weight matrix can be directly mapped to a single 128×128 crossbar array; if 256 > *n* > *128* and *m* $\leqslant$ *128*, the weight matrix can be mapped to the two crossbars at one column in a crossbar group; similarly, if *256* > *m* > *128* and *n* $\leqslant$ *128*, the weight matrix can be partitioned and separately mapped onto a row of two crossbars within a crossbar group. Our experiments shall show that the above mappings scheme can cover most three/four-layer multi-layer perception (MLP) topologies utilized in the simulated applications. In case of mapping an even larger MLP onto NCA, the full network may be partitioned into the sub-tasks within the above size ranges. If the total number of partitioned sub-tasks exceeds the availability of the crossbar arrays in the NCA, the MLP may be partitioned at software level and computed serially. Multiple target codes may be executed by the same NCA simultaneously as long as the crossbar groups/arrays are available. Note that the connection weights of the matrix do not change during NCA computations.

B. *Mixed-signal Interconnection Network*

The signal transmission between different crossbar groups can be realized in either digital or analog form. Digital signal transfer has good reliability in long distance transmission and supports high-frequency operations. However, as the computation of memristor crossbar arrays is in analog form, digital-to-analog/analog-to-digital (DA/AD) conversions are required at the interface of crossbar arrays and routers, which inevitably degrades the signal precision and results in significant area and power overheads. The small footprint of the memristor crossbars limit the data communication distance, e.g., within 0.53mm in our design, making it possible to transfer signals in also analog form. Moreover, the impact of signal distortion generated during the analog signal transmission on computation reliability can be tolerated by the intrinsic high fault resistance of ANN algorithms. The interconnection solution we propose is called "M-Net", which is a mixed-signal interconnection network. M-Net maintains the data in analog form while it transfers the control and routing information in digital form so as to simplify the synchronization and communication between CPU and NCA.

Figure 4 (a) shows the group router design. Its analog data path consists of input buffers and data multiplexer/switches. Each input port can receive up to 64 analog signals corresponding to a set of the inputs/outputs of a crossbar array, referred as a packet. The digital controller of a router is shown in Figure 4 (b).

V. Experiments

We investigate the performance of our proposed NCA by comparing with other ANN accelerators on multiple machine learning tasks [14]. First, we construct a digital neural processing unit (D-NPU) which adopts the NCA topology but replaces crossbar arrays and M-net with digital processing elements (PEs) [8] and a digital interconnection network DNet. To perform a fair comparison, the input/output FIFO and weight cache of each PE are scaled up to match the computational capacity of a crossbar array. The PE latency and

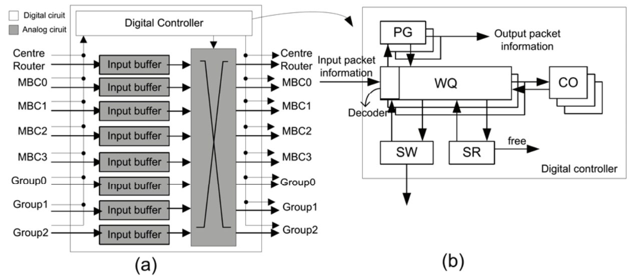

Figure 4. The mixed-signal router design (a) architecture (b) digital controller [19].

power are extracted from a VerilogHDL model synthesized with 65nm library using VCS and Design Compiler. In addition, to study the efficacy of MNet, we also construct an alternative design by solely replacing the M-Net in NCA with D-Net. The configuration of D-Net is estimated in booksim to offer the similar transmission capacity as M-Net. Meanwhile, DAC/ADC pairs are required at the interface of each router for the frequent DA/AD conversions before/after any memristor crossbar-based computation. Figure 5 compares the performance, energy efficiency, and classification rate of three ANN accelerator designs: DNPU, MBC+D-Net, and our proposed NCA. Here, the energy efficiency is defined as the inverse of system energy consumption. The performance and energy efficiency are normalized to those obtained from the baseline CPU execution. As shown in Figure 5 (a,b), our NCA has the highest speedup due to the higher computation ability than DNPU and the reduction of DA/AD conversions compared with MBCs+D-Nets. Figure 5 (c,d) compare the energy efficiency of all the designs, which demonstrates a similar trend as the performance results. Our NCA generally achieves more than 2× higher energy efficiency than that of MBC+D-Net due to the dramatically reduced DA/AD conversion overhead. Figure 9(e,f) compare the classification rate of all the designs. NCA shows slightly degraded

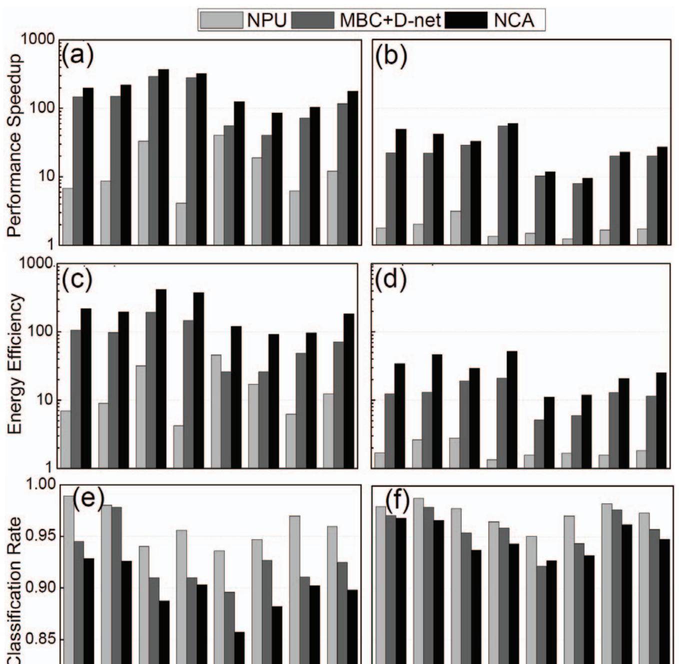

Figure 5. Comparisons of three ANN accelerators [19].

classification accuracy with other accelerators.

VI. CONCLUSION

In this work, we reviewed on-going research at EI lab on neuromorphic computing systems. A memristor crossbar based BSB model implementation has been demonstrated. Besides, a reconfigurable memristor-based NCA has also been depicted. Compared to a conventional CPU, our proposed NCA achieves on average 177.67× performance speedup and 184.71× energy reduction over the simulated benchmarks processed by implementing neural networks. The computation accuracy degradation is well constrained within a reasonably low range. The NCA can also support a variety of ANNs by properly reconfiguring the M-Net and guiding data routing among the memristor crossbar arrays.

ACKNOWLEDGMENT

We thank for the support from NSF CNS-1253424, NSF XPS-1337198, HP Lab Innovation Research Program, AFRL FA8750-15-2-0048, and DARPA D13AP00042.

REFERENCES

[1] P. Camilleri, M. Giulioni, V. Dante, D. Badoni, G. Indiveri, B. Michaelis, J. Braun, and P. del Giudice, “A neuromorphic avlsi network chip with configurable plastic synapses,” in International Conference on Hybrid Intelligent Systems, 2007, pp. 296–301.

[2] J. Partzsch and R. Schuffny, “Analyzing the scaling of connectivity in neuromorphic hardware and in models of neural networks,” Neural Networks, IEEE Transactions on, vol. 22, no. 6, pp. 919–935, 2011.

[3] B. Belhadj et al., “Continuous real-world inputs can open up alternative accelerator designs,” in ISCA, 2013, pp. 1–12.

[4] H. Esmaeilzadeh et al., \Neural acceleration for general-purpose approximate programs," in MICRO, 2012, pp.449-460.

[5] J. Gu et al., \Implementation and evaluation of deep neural networks (dnn) on mainstream heterogeneous systems," in APSys, 2014, p. 12.

[6] A. Schultz, “Collective recall via the brain-state-in-a-box network,” Neural Networks, IEEE Transactions on, vol. 4, no. 4, pp. 580–587, 1993.

[7] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, “The missing memristor found,” Nature, vol. 453, pp. 80–83, 2008.

[8] The mnist database," http://yann.lecun.com/exdb/mnist/.

[9] Uci machine learning," http://archive.ics.uci.edu/ml/.

[10] F. Alibart et al., \High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," Nanotechnology, vol. 23, no. 7, 2012.

[11] B. Belhadj et al., \Continuous real-world inputs can open up alternative accelerator designs," in ISCA, 2013, pp. 1{12.

[12] L. O. Chua, \Memristor-the missing circuit element," Circuit Theory, vol. 18, no. 5, pp. 507{519, 1971.

[13] M. Gustavsson, J. J. Wikner, and N. Tan, CMOS data converters for communications, 2000.

[14] S. O. Haykin, Neural Networks and Learning Machines.London: Prentice Hall, 2008.

[15] M. Hu et al., \Hardware realization of bsb recall function using memristor crossbar arrays," in DAC, 2012, pp.498{503.

[16] N. Jian et al., \A detailed and oxible cycle-accurate network-on-chip simulator," in ISPASS, 2013, pp. 86{96.

[17] K.-H. Kim et al., \A functional hybrid memristor crossbar-array/cmos system for data storage and neuromorphic applications," Nano letters, vol. 12, no. 1, pp. 389{395, 2011.

[18] B. Liu et al., “Digital assisted noise eliminating training for memristor crossbar based analog neuromorphic computing engine,” in DAC, 2013, pp. 1–6.

[19] X. Liu et al., “RENO: a high-efficient reconfigurable neuromorphic computing accelerator design,” in DAC, 2015, pp. 1–6.

[20] B. Liu et al., “Reduction and IR-drop compensations techniques for reliable neuromorphic computing systems,” in ICCAD, 2014, pp. 63-70.

[21] O. Temam, “A defect-tolerant accelerator for emerging high-performance applications,” in ISCA, 2012, pp. 356–367.