Harmonica: A Framework of Heterogeneous Computing Systems With Memristor-Based Neuromorphic Computing Accelerators

Xiaoxiao Liu, Mengjie Mao, Beiye Liu, Boxun Li, Yu Wang, *Senior Member, IEEE*, Hao Jiang, *Member, IEEE*, Mark Barnell, *Member, IEEE*, Qing Wu, *Member, IEEE*, Jianhua Yang, *Member, IEEE*, Hai Li, *Senior Member, IEEE*, and Yiran Chen, *Senior Member, IEEE*

***Abstract*—Following technology scaling, on-chip heterogeneous architecture emerges as a promising solution to combat the power wall of microprocessors. This work presents *Harmonica*—a framework of heterogeneous computing system enhanced by memristor-based neuromorphic computing accelerators (NCAs). In Harmonica, a conventional pipeline is augmented with a NCA which is designed to speedup artificial neural network (ANN) relevant executions by leveraging the extremely efficient mixed-signal computation capability of nanoscale memristor-based crossbar (MBC) arrays. With the help of a mixed-signal interconnection network (M-Net), the hierarchically arranged MBC arrays can accelerate the computation of a variety of ANNs. Moreover, an inline calibration scheme is proposed to ensure the computation accuracy degradation incurred by the memristor resistance shifting within an acceptable range during NCA executions. Compared to general-purpose processor, Harmonica can achieve on average 27.06× performance speedup and 25.23× energy savings when the NCA is configured with auto-associative memory (AAM) implementation. If the NCA is configured with multilayer perception (MLP) implementation, the performance speedup and energy savings can be boosted to 178.41× and 184.24×, respectively, with slightly degraded computation accuracy. Moreover, the performance and power efficiency of Harmonica are superior to the designs with either digital neural processing units (D-NPUs) or MBC arrays cooperating with a digital interconnection network. Compared to the baseline of general-purpose processor, the classification rate degradation of Harmonica in MLP or AAM is less than 8% or 4%, respectively.**

***Index Terms*—Heterogeneous system, memristor, neuromorphic computing.**

Manuscript received October 7, 2015; revised December 22, 2015; accepted January 15, 2016. Date of current version June 28, 2016. This work is supported in part by NSF 1337198, NSF 1253424, AFRL FA8750-15-2-0048, HP Lab Innov. Res. Pgm, and approved for public release by AFRL on 03/04/2015, case number 88ABW-2015-0833. This paper was recommended by Associate Editor E. A. B. da Silva.

X. Liu, M. Mao, B. Liu, H. Li, and Y. Chen are with the Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA 15260 USA (e-mail: xil116@pitt.edu; mem231@pitt.edu; bel34@pitt.edu; hal66@pitt.edu; yic52@pitt.edu).

B. Li and Y. Wang are with the Department of Electronics Engineering, Tsinghua University, Beigjing 100084, China (e-mail:lbx13@mails.tsinghua.edu.cn; yu-wang@mail.tsinghua.edu.cn).

H. Jiang is with the Department of Electrical Engineering, San Francisco State University, San Francisco, CA 94132 USA (e-mail: jianghao@sfsu.edu).

M. Barnell and Q. Wu are with Air Force Research Laboratory, Rome, NY 13441 USA (e-mail: mark.barnell.1@us.af.mil; qing.wu.2@us.af.mil).

J. Yang is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003 USA (e-mail: jjyang@umass.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2016.2529279

I. Introduction

RECENTLY, heterogeneous architecture has become a promising solution to conquer the challenges of supply voltage scaling, off-chip communication bandwidth, and application parallelism in homogeneous multi-core system [1]. Several off-chip accelerators, including traditional ASIC, FPGA, and GPU, have been well studied for cooperating with general-purpose processors [2]–[4]. Generally, ASIC provides the highest computation efficiency and FPGA offers the most flexible reconfigurability. GPU is a balanced solution between these two metrics though its applications are often associated with complex control flows and special programming models.

Besides off-chip accelerators, many practices [5]–[8] were also conducted to integrate general-purpose CPU cores with processing elements that are designed to accelerate the execution of some special codes (called *target codes*), e.g., the codes producing approximated results. Many target codes of approximate computing (e.g., approximated calculation, rendering methodology, and statistical representation) have been identified in a large variety of applications such as pattern recognition, computer vision, data mining, signal processing etc. [9], [10]. *Artificial neural network* (ANN) can be also considered as one kind of approximate computing with high adaptivity to many high-performance applications [9]. The inherent resilience to soft and hard errors in computation makes ANN a promising solution to conquer the aggravated system reliability issue under the highly scaled technology nodes [11]. Software-based ANN realizations, however, are often associated with extremely high hardware cost required by emulating the complex connections in the neural network.

The rediscovery of memristor [12] motivates an exciting approach of implementing neuromorphic systems, which denotes the VLSI realization of ANN computation. Compared to the design of traditional CMOS-based digital and analog neuromorphic accelerators [10], [13], the similarity between the programmable resistance state of memristors and the variable synaptic strengths of biological synapses dramatically simplifies the structure of neural network circuits [14].

In this work, we propose ***Harmonica***—a novel framework of heterogeneous computing systems with on-chip memristor-based *neuromorphic computing accelerators* (NCAs), aiming at the acceleration of ANNs and learning computations. Nanoscale *memristor-based crossbar* (MBC) arrays [15] are

utilized to represent perceptron network in NCA development. Unlike the spike-based computations where the data is represented by the pulse signals with different frequencies and amplitudes [16], our design adopts a hybrid data presentation: the computation within MBCs and the signal transmission among MBCs are conducted in analog form while the control information is maintained in digital form. In this work, we assume the training process of MBC arrays is performed offline. However, to suppress the accuracy degradation incurred by memristor resistance shifting during executions, a low-cost inline calibration scheme [17] is applied.

Harmonica offers a fast, cost-efficient and fault-tolerant ANN computation platform complementing the computations of CPU cores. Compared to the existing works of approximate computing units and digital ANN accelerators, the key differentiation of Harmonica can be summarized as:

- ***A novel mixed-signal NCA*** is built based on the emerging memristor technology, offering orders of magnitude performance and power efficiency improvement w.r.t. general-purpose microprocessors;
- ***A hierarchical MBC array structure*** that can be easily configured to different ANN topologies is designed;
- ***A mixed-signal interconnection network (M-Net)*** is proposed to conduct the data migration in analog form among the MBC arrays. The attempt is to minimize the signal conversion between digital and analog forms, and therefore reduce the performance and power overheads;
- ***An inline calibration scheme*** is also developed to ensure the computation accuracy degradation incurred by the memristor resistance shifting within an acceptable range;
- ***The performance and accuracy of NCA*** are thoroughly analyzed by examining various design parameters.

A set of prevailing ANN benchmarks is adopted in the evaluation of Harmonica. *Multilayer perception* (MLP) and *auto-associative memory* (AAM) are used to represent two typical tradeoffs between the computation accuracy and performance. As shown in experimental results, Harmonica with AAM (MLP) implementation can achieve on average 27.06× (178.41×) performance speedup and 25.23× (184.24×) energy saving over the seven selected ANN applications, compared to the baseline CPU. Furthermore, the comparisons with a *digital neural processing unit* (D-NPU) [13] and a conventional mixed-signal accelerator design show Harmonica performs the best balance among performance, power consumption and computing accuracy. Besides, the inline calibration scheme successfully suppresses the influence of memristor resistance shifting on the NCA computation accuracy with less than 0.41% (0.86%) performance overhead in MLP (AAM) implementation.

II. Preliminary

A. Artificial Neural Network (ANN)

There are two canonical ANNs are considered in this work, including 1) MLP with high computation efficiency and 2) AAM with high computation accuracy.

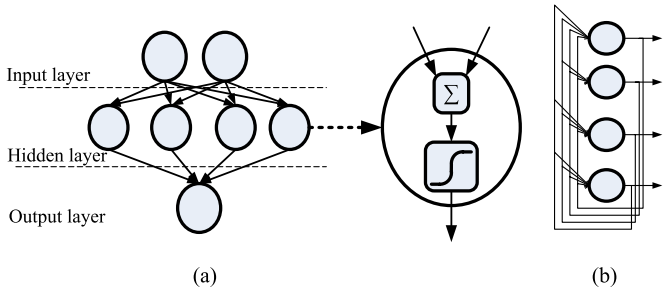


Fig. 1. (a) A 3-layer MLP. (b) A 1-layer AAM with 4 neurons.

MLP is a type of feedforward ANNs that have been widely studied in the research of approximate computing [18]. MLP maps a set of input data to its outputs through multiple layers of nodes in a directed graph. Every layer in the MLP is fully connected to the next layer. An example of 3-layer MLP is shown in Fig. 1(a): The input nodes collect and convey the input bits to the next layer through many weighted connections. A weighted connection (or *synapse*) is associated with a preset weight by which the carried signal can be modulated. Except for the input nodes, each of other nodes in the network represents a *neuron* with a nonlinear activation function, e.g., a sigmoid function $f(x) = 1/1 + e^x$ on the sum of all the signals that the node receives.

AAM is often utilized for pattern recognition and completion [19]. A Hopfield network acting as an AAM is illustrated in Fig. 1(b). Each pair of neurons in the network are linked through a weighted connection. An input vector will go through the network iteratively and converge to the closest version of the vector pattern, offering good immunity to noises or other randomness in the computation. Generally, the non-iterative MLP executes faster than the iterative AAM; but AAM is much more dependable because of its inherent fault tolerance characteristic.

As a major function of ANN, training decides the weight of each connection and makes the ANN able to properly respond to unseen data with desired outputs. In this work, back-propagation and delta rule [20] are adopted to perform the training of MLP and AAM. Our architecture-level contributions aims at the ANN testing/computation process by assuming the NCA has been trained by supervised algorithms for specific applications. The configuration of NCAs does not require further modification during the computation except for the inline calibration, which will be discussed in Section IV-C.

B. Memristor and Memristor-Based Crossbar (MBC)

As defined and predicted by Professor Leon Chua, memristor is regarded as the 4th fundamental electrical element [21]. The resistance (*memristance*) of a memristor is uniquely determined by the electric charge/flux through the device. Theoretically, the resistance of a memristor can be tuned to any state between the lowest and highest conductance limits by applying voltages with different strength and/or duration [22], [23]. Compared to other popular nonvolatile memory devices, e.g., phase change memory [24] and spin-transfer-toque magnetic-tunneling-junction [25], memristor offers the highest integration density (only $4F^2$), the largest $\mathbf{R}_h/\mathbf{R}_L$ ratio ($\approx$800 based on [26]),

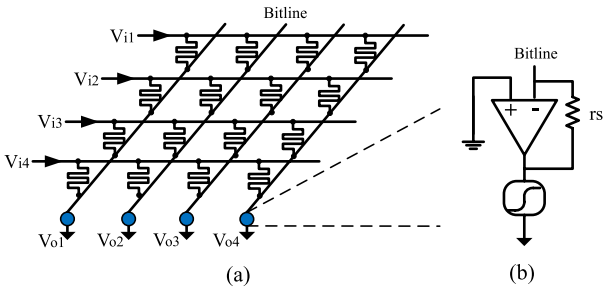


Fig. 2. (a) A 4 × 4 MBC array. (b) The neuron logic.

and bipolar programmability that greatly improve the training efficiency of the NCA. A promising multi resistance state memristor device has been recently demonstrated to perform 7-bit programming resolution with a carefully designed tuning mechanism [27].

The unique property of recording historical profile of electric excitation makes memristor behave similar to the biological synapse [12], [28]. This similarity inspired many studies on realization of synapse with memristor device. For example, the capability of being programmed by spike timing dependent plasticity (STDP) learning rule allow researchers to use memristor to implement a spiking networks [12].

Fig. 2 shows the structure of an MBC that fully connects two adjacent layers of neurons in an MLP. The relationship between input voltages $(\mathbf{V}_i)$ and output voltages $(\mathbf{V}_o)$ are define by the resistances of all memristors in an MBC, and can be can be described by

$$\mathbf{V}_o = \mathbf{C} \times \mathbf{V}_i. \tag{1}$$

Here $\mathbf{C}$ is the conductances of all memristor devices, which is often defined as "weight matrix." In actual hardware, the operation defined by (1) will be affected by operational noises, e.g., memristor resistance variation and voltage drop [29] on the connection wires. In order to implement an MLP with expected feed-forward function, a series of MBCs need to be connected. A large amount of linear algebra can be then performed in parallel as there is little dependency between the data within the same layer. In [30], Hu *et al.* proposed using MBCs to perform vector-matrix multiplication in analog form with minimum impact from sneak path. In this work, similar design is adopted to implement the computing core of the proposed ANN accelerator.

III. Harmonica System Overview

Kuon *et al.* forecasted the rising of analog neuromorphic computing circuitry because of its great potential in energy efficiency and computation density [31]. In Harmonica, the acceleration of ANN computation is performed in a mixed-signal *neuromorphic computing accelerator* (NCA) based on the MBC structure presented in Section II-B. The synapse weights in an ANN are represented by the resistances of the memristors. Routers are introduced to connect the MBCs and conduct the topological reconfiguration of the NCA (e.g., MLP and AAM).

Fig. 3 illustrates the proposed Harmonica architecture. In each processor core, the general-purpose pipeline is augmented with a NCA. Within the territory of the NCA, the computation as well as the data transportation all remain in analog form.

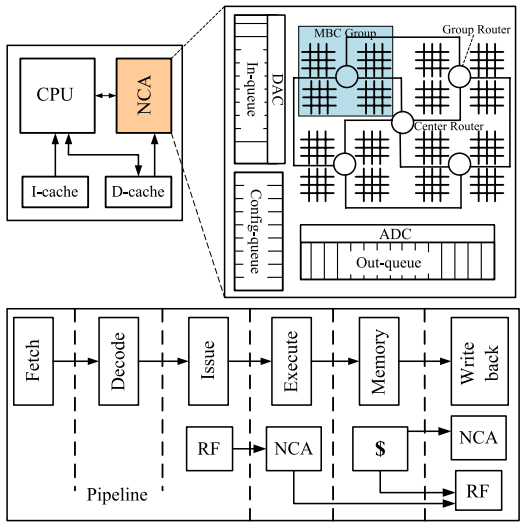


Fig. 3. Overview of Harmonica architecture.

A *mixed-signal interconnection network* (M-Net) is responsible for the routing of the analog computational data and the digital control signal. *Digital-analog/analog-digital* (DA/AD) conversion is performed on the data from/to the general-purpose pipeline at the boundary of the NCA to accommodate different computation formats. Such a design minimizes the performance and energy overheads of DA/AD conversions and maximizes the reconfigurability of the NCA.

A compilation flow similar to [13] is utilized to generate a NCA-aware binary of which the target ANN code is executed in the NCA. Based on the characteristics and complexity of the target codes, the ANN implementation topology, including the number of layers and the number of neurons at each layer, is decided offline. The initial training of the NCA for specific applications is also completed offline. Considering that the resistance states of the memristors could gradually drift during the NCA computation [32], we propose an inline calibration scheme to periodically refresh the MBCs and calibrate the computation accuracy of the NCA by finely calibrate the MBCs with a subset of the training vectors.

IV. The NCA Architecture

Memristor-based *neuromorphic computing accelerator* (NCA) is the key component enabling the high performance and energy efficiency of Harmonica. This section presents the hardware design details of the NCA. We first propose a hierarchical structure of reconfigurable MBC arrays to support both MLP and AAM implementations. The necessity of introducing *mixed-signal interconnect network (M-Net)* is discussed and the implementation details are also given. An inline calibration scheme is then proposed to ensure run-time execution accuracy of the NCA. Finally, the interaction between the CPU and the NCA is discussed at the end of this section.

A. Hierarchical Structure of MBC Arrays

Fig. 3 shows the hierarchical MBC array structure in the NCA. We adopt the *metamorphous centralized mesh* (MCMesh) topology to build the interconnection network for

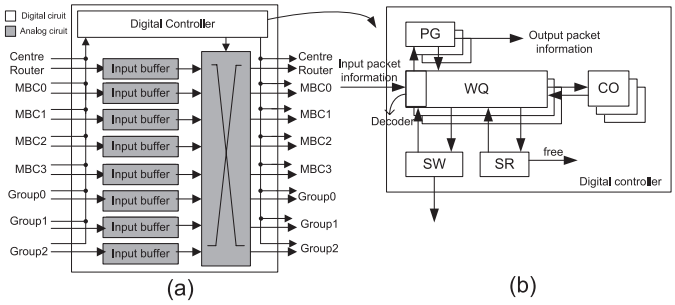


Fig. 4. The mixed-signal router design: (a) Architecture. (b) The digital controller.

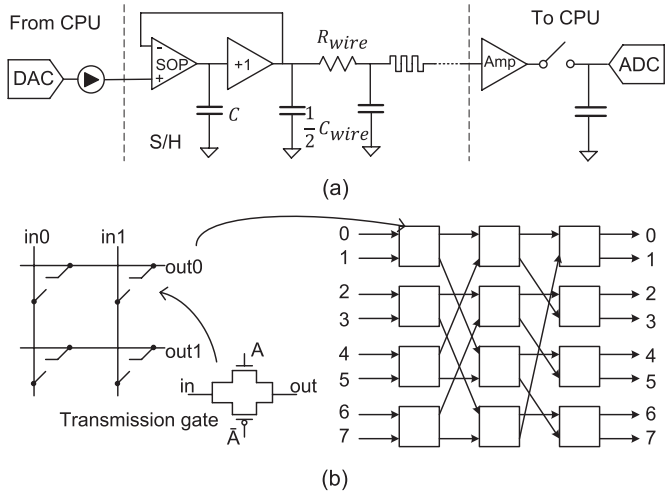


Fig. 5. The analog component design in the mixed-signal router: (a) The transmission path. (b) The crossbar-based multiplexer.

supporting data migration among MBC arrays as it has less cost than most of the popular network topologies in multicore systems [33]. NCA consists of four MBC groups, each of which has four MBC arrays connected through a group router. An MBC array consists of four sub-arrays to present the 4 combinations of the multiplication of the signed input signal and the singed weight as described in [34]. In this work, the optimized MBC design has 64 rows and 64 columns. As we shall show in Section VI-D, such a design offers a good compromise between performance and reliability. Furthermore, since 80% of learning applications have no more than 60 neurons in the input layer [11], this MBC design is sufficient for the majority of ANN applications. Meanwhile, with the interconnection network, larger applications can be supported by being partitioned into multiple MBC arrays for execution.

Without losing generality, we use a connection matrix $M_{n\times m}$ to explain how a connection matrix can be mapped into the MBC arrays. Here n and m represent the numbers of neurons in the input and output layers of the connection matrix, respectively. If $\max(n, m) \leq 64$, $M_{n\times m}$ can be directly mapped to a 64×64 MBC array; if $64 < n \leq 128$ and $m \leq 64$ or if $64 < m \leq 128$ and $n \leq 64$, $M_{n\times m}$ can be mapped to two MBC arrays; an even larger $M_{n\times m}$ need be partitioned into more MBC arrays in different MBC groups.

B. Mixed-Signal Interconnection Network (M-Net)

1) Digital, Analog, or Mixed-Signal: The signal transmission in the NCA can be conducted in either digital or analog form. Digital signal can support very high-frequency data transfer. However, as the computation performed by MBC arrays is in analog form, DA/AD conversions are necessary at the interface between each MBC array and the connected router if the signal transmission is in digital form. Such a design inevitably harms the signal precision and introduces significant area and power overheads. In our NCA design, the small footprint of MBC arrays keeps the data communication distance less than 0.53 mm, which indeed allows the data to be transferred in analog form. Moreover, the impact of signal distortion generated during the signal transmission can be tolerated by the high fault resistance of ANN.

We design a mixed-signal interconnection network (M-Net) to assist the computation and data migration in the MBC arrays. In M-Net, the computational data is maintained in analog form, while the control and routing information is transferred in digital form to simplify the communication and synchronization between the CPU and the NCA. More specifically, the signal communication is conducted through routers, each of which is divided into digital control logic and analog data path.

Fig. 3 also shows the centralized hierarchical MBC array architecture where the data communication is performed at both inter-group and intra-group levels. The *central router* connects to the CPU and all the ***group routers***. Each group router talks to the four local MBC arrays within the group, three other group routers, and the central router. Such a centralized scheme maximizes the number of parties that each router communicates with, minimizes the effective communication distance and the hop count, mitigates the bottleneck effect of the central router, and simplifies the control complexity.

2) Router Design: The group router design is depicted in Fig. 4(a). The analog data path [shown in Fig. 5(a) and (b)] is composed of 8 input/output ports, input buffers and data multiplexer/switches. The 8 input/output ports are connected with 4 local MBC arrays, 3 other group routers, and the CPU, in which 64 analog signals in each port corresponding to a set of data as one package. We adopt a *switched-op-amp* (SOP) based *sample-and-hold* (S/H) circuit (see Fig. 5(a) [35]) as an analog buffer to hold the analog data until the data is ready to be transferred to the next destined MBC array or router. Such a design substantially minimizes the nonlinear distortion of the stored analog data caused by charge injection and clock feedthrough error, maintaining a good signal quality [35]. More discussions are be found in Section V-A.

Fig. 4(b) shows the digital controller design in a router. Like traditional router in CMesh NoC, the routers in NCA are responsible for both data transmission and routing info processing. A *work queue* (WQ) is introduced to monitor the MBC array computation status and produce the routing info for each data packet. It also controls the routing path configuration in the multiplexer through a *switch allocator* (SA). Each WQ entry is associated with a multi-bit *computing counter* (CO) to monitor the computation status of a local MBC array by counting the number of the executed loops. We also introduce

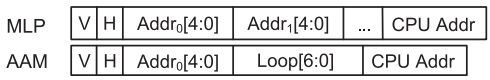


Fig. 6. Routing information format.

a multi-bit *computing counter* (CO) to count the computation time for an MBC array, a *packet generator* (PG) to produce the routing information for the next data transmission, and a *Status recorder* (SR) to broadcast the status of the data path to the connected router.

The central router design is similar to that of the group router except that the central router is only responsible for establishing the data paths between the CPU and the four group routers. Although the group routers work independently, all the MBC arrays can perform computation simultaneously.

3) Routing Management: We design a special routing information package for data routing in NCA to support a variety of ANN implementations (e.g., AAM and MLP). The routing information package consists of 1-bit valid bit (V), 1-bit routing field (H), address field (Addr_i), and looping field (Loop). Since a NCA consists of 4 MBC groups and each group has 4 MBC arrays, the address field contains 5 bits: $\text{Addr}_i[1:0]$ represents the MBC group, $\text{Addr}_i[3{:}2]$ identifies the MBC array within the group, and $\text{Addr}_i[4]$ indicates CPU or NCA. The looping field contains 7 bits to support up to 128 loops corresponding to the requirement of the seven ANN benchmarks.

Bit H denotes the type of ANN implementations (MLP or AAM) and the composition of routing information. The MLP implementation only has address field. The address field indicates the path of the data travelling in the NCA when MLP implementation is selected. The AAM implementation requires both address and looping fields to guide the destined MBC array and the number of computation looping, respectively. *CPU address* is always at the end of the routing information package, completing the data transmission in NCA.

C. MBC Inline Calibration

Ideally, MBCs are programmed in training process by applying proper electrical current/voltage. During NCA operations, only small computation signals go through the memristors. However, such small signals can still disturb the memristor resistance states, resulting in the deviation of synapse weights from the target values. Most importantly, such memristor resistance shifting accumulates over time and eventually leads to the degradation of NCA computation accuracy.

We studied the memristor resistance shift when executing benchmark *connect-4* in the NCA with MLP and AAM implementations, respectively. Here the memristor resistance shift is measured by the relative deviation, which denotes the ratio between the resistance change and the originally trained value. The simulation results are summarized in Fig. 7, where $x-$axis represents the number of NCA runs and $y-$axis represents the accumulated percentage of the memristors on the crossbar with a relative resistance deviation $> 10\%$. As we can observe, the time-varying inputs in MLP implementation randomize the memristor resistance shift and slow down its accumulation. In AAM implementation, however, the iterative computations

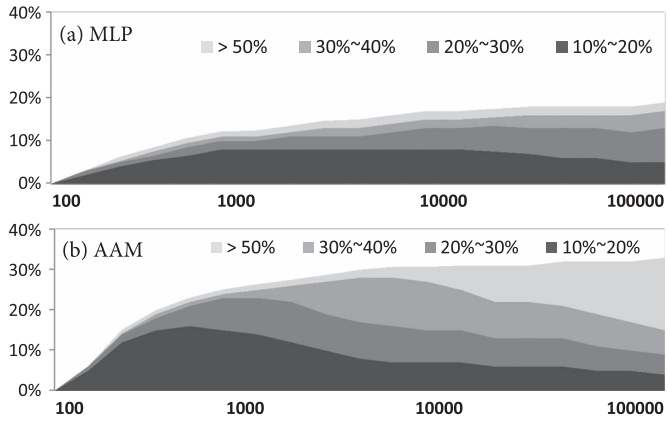


Fig. 7. The memristor resistance drifting when executing *connect-4* in (a) MLP or (b) AAM implementation.

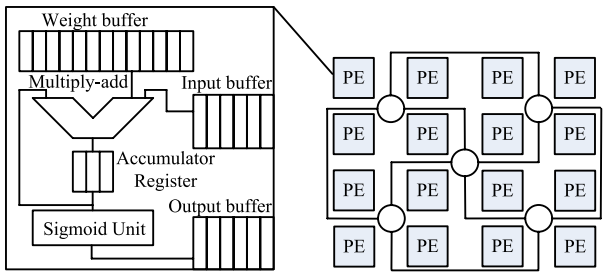


Fig. 8. A D-NPU design built with digital PEs in [13].

under the consistent inputs cause a fast accumulation of the memristor resistance shift.

We propose an inline calibration scheme as follows: if the resistances of an MBC are too far from the originally trained values to accurately perform the computation, a set of training vectors will be applied to finely tune the resistance of the MBC back. Although the memristor resistance is not easy to converge to the target level which requires iterative programming with feedback control [36], the inline calibration can be conducted anytime between executions of two NCA instructions and hence, has no impact on the continuity of the NCA operation. During a calibration process, the NCA outputs are used for only fine-training purpose (e.g., utilizing Delta rule) and will not be sent to the CPU.

On the one hand, a reasonable calibration interval shall be maintained to avoid unnecessary MBC fine tuning. On the other hand, prolonging the interval between two calibrations does not necessarily reduce the overall calibration overhead: a longer interval potentially raises the memristor resistance shift and hence increases the required training time. More relevant discussions can be found in Section VI-C.

D. Interaction Between CPU and NCA

We extend the *instruction set architecture* (ISA) by adding special NCA instructions to control the NCA in Harmonica. A NCA-aware compiler is developed to compile the target codes into two types of NCA instructions: 1) The *NCA I/O instructions* that are used to supply inputs to the NCA and also collect its outputs; and 2) the *NCA configuration instructions*

TABLE I
THE EXTENDED NCA INSTRUCTIONS IN ISA

Instruction	Type	Description
$setp$ reg	Configuration	Place the routing information stored at register reg to central router.
$movd$ #(reg)	I/O	Load the data from memory to NCA.
$launch$	Configuration	Notify the central router to start transmitting
deq reg	I/O	Dequeue the head data of Out-queue and write it to register reg.

TABLE II
THE DESIGN PARAMETERS OF NCA COMPONENTS

Memristor						
R_L=200Ω, R_H=160KΩ, V_{th}=2V						
MBC Array & M-Net						
	Op amp	Network	Sigmoid	MBC	DAC	ADC
Power	100μW	0.72μW	10μW	0.69μW	5.2mW	3.8mW
Speed	0.60ns	4.2ns	0.24ns	3ns	333MHz	333MHz
Area Estimation						
	NCA area (mm^2)	NoC (mm^2)			DAC/ADC (mm^2)	MBC (mm^2)
		Input/output	Channel	Control		
M-Net	0.943	0.598	0.014	0.252	0.072	0.007
D-Net	1.793	0.268	0.065	0.301	1.152	0.007

that setup data path through MBC arrays. Detailed definitions of these extended NCA instructions are listed in Table I. The parameters of ANN, including the number of inputs/outputs, layers and neurons, are imported to the compilation and assist the generation of NCA configuration instructions.

The NCA works as a complementary functional unit to the CPU and accelerates ANN-relevant executions of the target codes. Fig. 3 illustrates how NCA interacts with the CPU at different pipeline stages: At issue stage, setp instruction reads the register file and sets up the data routing in the NCA. launch instruction fires the NCA execution at execute stage. deq instruction retrieves the output data from the NCA at execute stage and writes them to the register files at write back stage. $movd$ instruction feeds the input data to the NCA at write back stage.

All the interactions between the NCA and the CPU go through the three FIFO queues of the NCA, which are also shown in Fig. 3: *In-queue* buffers the data fed by movd instruction; *Out-queue* buffers the data that are produced by the NCA and wait to be fetched by deq instruction; *Config-queue* buffers the routing information given by setp instruction. The data popped from *In-queue* are converted to analog signals before being routed to the destined MBC array. Similarly, the output analog signals from the NCA should be converted to digital form before being captured by *Out-queue*.

V. EXPERIMENTAL METHODOLOGY

A. Circuit Level Implementation and Simulation

The adopted Verilog-A Memristor model is based on the device parameters from [26], which have been scaled to 65 nm technology according to the resistance and area relation in [37]. The memristor material is W/SiGe/a-Si/Ag and the read current applied on the memristor is 2 A. All the NCA circuit components, including MBC, analog buffer, switch, sum amplifier, sigmoid circuit, etc., are designed with SMIC 65 nm technology [38]. We use a 4-bit flash *analog-digital converter* (ADC) and a 4-bit current steering *digital-analog converter* (DAC) [39] to achieve a fast data transfer. The resolution is decided by the requirement of the selected benchmarks. As depicted in Fig. 5(a), an MBC array receives input data from the DAC and sends the computation result to the ADC. The DAC at the input side is coupled with a cascoded current amplifier to boost the output impedance. At the output side of the MBC array, a signal passes through an *amplifier* (Amp) and a *sample-and-hold* (S/H) before reaching the ADC. The Amp boosts up the input signal to match the ADC input window and performs *correlated-double-sampling* (CDS) to mitigate the DC offset caused by mismatch [40], while the S/H ensures a stable input during the analog-to-digital conversion. The area of all the analog circuits are extracted from Cadence Virtuoso [41] simulations while that of the digital component is estimated based on *Booksim* and summarized in Table II. As most of the NCA is occupied by routers, we build an analog signal transmission model to simulate the longest transmissible distance between the routers. The result shows that a voltage swing between 0 V and 1 V is safe to be transferred from one output port of a router to one input port of a connected router in 0.5 ns, after considering signal fluctuations. We also consider the possible noises produced in NCA, 1/f noise generated in the amplifier, thermal noise produced in the memristor and the amplifier, and quantization noise caused by the 4-bit ADC. After evaluations, we find the quantization noise, within 18 mV, dominates in these three types of noises and it is still negligible compared to the resolution of the 4-bit DAC/ADC, 62.5 mV. The device mismatch can be calibrated by a predetermined look-up table like [42]. In addition, the signal distortion introduced by MBC arrays is assumed to follow a normal distribution. The detailed discussion of the signal distortion impact can be found in Section VI-B.

Monte-Carlo simulations are run for reliability analysis by assuming both the memristor resistance and the analog inputs follow normal distributions. Since the initial memristor resistance of an MBC sample is decided by the offline training, it is fixed in each Monte-Carlo simulation. Contrarily, the signal fluctuation is generated on-the-fly during the NCA execution.

B. Benchmarks

As shown in Table III, seven representative ANN benchmarks are selected in our experiment.[1] All the selected benchmarks can be implemented using MLP or AAM models. The algorithm execution quality is measured by the classification rate. The implementation details and the initial training errors of the selected benchmarks using MLP and AAM model are also presented in Table III. These benchmarks naturally come with training and testing inputs. Hence, we are able to further divide the training vector into an actual training set and a so-called validation set, which is used to evaluate the quality of a network. ANN topology for each application is optimized based on FANN library [46] by comprising training time, computation accuracy, and network size. The enhanced device variation and signal noise aware MBC training scheme [20] is utilized to ensure training robustness. The *mean square error* (MSE) is applied to evaluate the reliability of the NCA.

[1]The image of MNIST is compressed from 28 × 28 pixels into 8 × 8 pixels and the gray scale is reduced from 256 to 16.

TABLE III
THE DESCRIPTION AND IMPLEMENTATION DETAILS OF THE SEVEN SELECTED BENCHMARKS

Benchmark	Description	MLP			AAM	
		Training error	Topology	MBC array usage	Training error	MBC array usage
cancer [43]	breast cancer diagnose	0.02%	36→16→2	2 arrays in 1 group	0.07%	2 arrays in 1 group
connect-4 [44]	connect-4 game	0.02%	42→30→3	2 arrays in 1 group	0.08%	3 arrays in 1 group
gene [43]	nucleotide sequences detection	0.09%	120→100→3	6 arrays in 2 groups	0.03%	12 arrays in 3 groups
lymphography [44]	lymph diagnose	0.05%	29→19→4	2 arrays in 1 group	0.02%	4 arrays in 1 group
MNIST [45]	digit recognition	0.35%	64→128→32→10	5 arrays in 2 groups	0.02%	10 arrays in 3 groups
mushroom [43]	poisonous mushroom discrimination	0.01%	125→32→2	3 arrays in 1 group	0.01%	8 arrays in 2 groups
thyroid [43]	thyroid diagnose	0.15%	21→32→3	2 arrays in 1 group	0.11%	3 arrays in 1 group

TABLE IV
THE SIMULATION PLATFORMS

CPU	***CPU core***	1.1GHz, OOO 2-issue(up to 1 mem and 1 FP), 20-stage pipeline, gshare branch predictor, 256-entry BTB, global history length 14, 20-cycle branch misprediction penalty, 256-enrty ROB
	Cache & Memory	32KB L1ICache, 4-way, 1 bank, 4-port, 1-cycle latency, 64B line 32 KB L1DCache, 4-way, 4-bank, 4-port, 2-cycle latency, write back, 64B line 256 KB L2, 8-way, 4-bank, 1 R/W port, 8-cycle latency, write back, 64B line, 2 MB L3, 16-way, 8-bank, 1 R/W port, 16-cycle latency, 64B line, write back, 128 MSHRs 4GB main memory, fixed 50-cycle latency
NCA	***Computing Units***	4 MBC groups, 4 MBC arrays/group, 4 MBC/array, MBC size: 64×64
	IO	64×4-bit In-queue/Out-queue, 128×64-bit Config-queue, 64 parallel DACs, 64 parallel ADCs
	Network (M-Net)	Mixed-signal CMesh, 333 MHz for digital control, 4 group routers, 1 central router
D-NPU	***Computing Units***	16 digital PEs, Input/Output Buffer 64×4-bit, Weight Cache 4096×4-bit, Sigmoid Unit LUT 512×4-bit, 4-bit Multiply-add unit
	IO	64×4-bit In-queue/Out-queue, 128×64-bit Config-queue
	Network (D-Net)	Digital CMesh, 1.332 MHz, 64-bit datapath, 4 group routers, 1 central router

C. Architecture Level Simulation Setup

We add a cycle-accurate NCA module in *MacSim* [47] and configure the CPU as Intel Atom [48]. The NCA-supported functions within a target code are identified and translated to NCA instructions. During trace generation, the modified PIN tool generates the simulation trace by replacing the NCA-supported functions with the corresponding NCA instructions according to the selected ANN topology in the specific application. For the benchmarks that we evaluated, averagely 99% of execution time is spent on running the target code. Thus, we only consider the execution time of the target codes instead of the whole program. All the parameters are depicted in Table IV.

We obtain the energy consumption of NCA by recording the execution during NCA computation and calculating based on the circuit level simulation result. The CPU's energy consumption is generated by *McPAT* [49]. The data traffic and the power consumption of both the M-Net and the D-Net are simulated by a modified *booksim* simulator [50].

D. Implementation of Other Design Alternatives

We also compare the NCA-based design with other ANN-specific accelerator designs. A digital accelerator design—*processing elements* (PEs) from [13] is evaluated by scaling the input/output/weight buffer of each PE to the level of an MBC array. Through a digital NoC (namely, *D-Net*) that has the same throughput as our proposed M-Net, 16 PEs are connected to construct the *digital neural processing unit* (D-NPU). Table IV shows the configuration of the D-NPU.

We also conduct an alternative design to explore the efficacy of M-Net by connecting MBC arrays with D-Net instead of M-Net. The MBC arrays remain as the computing units. D-Net keeps the same topology and function as M-Net by transmitting both data and control signals in digital format. To minimize the design cost of data bus while maintaining the same bandwidth, digital data can be packed and transmitted at a higher frequency. The evaluation in *booksim* [50] shows that operating the D-Net with input buffers at 1.332 GHz offers the similar transmission capacity as M-Net. The boundary of digital and analog domains moves from CPU ↔NCA in Harmonica to D-Net ↔MBC arrays in such a "*MBCs+D-Net*" design. DA/AD conversions are frequently performed before/after every MBC computation and become indispensable. Compared to the M-Net, digital transmission on the D-Net suppresses signal precision loss and simplifies the router design. However, the increased number of DA/AD converters dramatically increases the design area and power consumption of the non-computing parts, as illustrated in Table II.

VI. EXPERIMENTAL RESULTS

A. MBC Training Effort

The computation accuracy and energy consumption of the NCA are greatly influenced by the precision of the input signal and the training effort which can be measured by the size of training data set used in MBC array training. The resolutions of the computation data are naturally provided in the selected benchmarks and all less than or equal to 4-bit. Thus, we fix the DAC/ADC resolution to 4-bit and focus on the impact of the training effort in the following evaluations.

For a specific benchmark, the computation accuracy can be improved by increasing the size of training data set. However, the computation accuracy will saturate when the number of the training data reaches a threshold, i.e., the saturated training data set size. We compare the computation accuracy of MLP and AAM under different training efforts as shown in Fig. 9(a) and (b), respectively. Here 100% training effort denotes the case that the saturated training data is used and all the classification rates are normalized to the ideal value achieved by the CPU with 32-bit floating point precision. As we can see, applying 100% training effort will produce a normalized classification rate very close to the ideal case. The classification rate decreases as

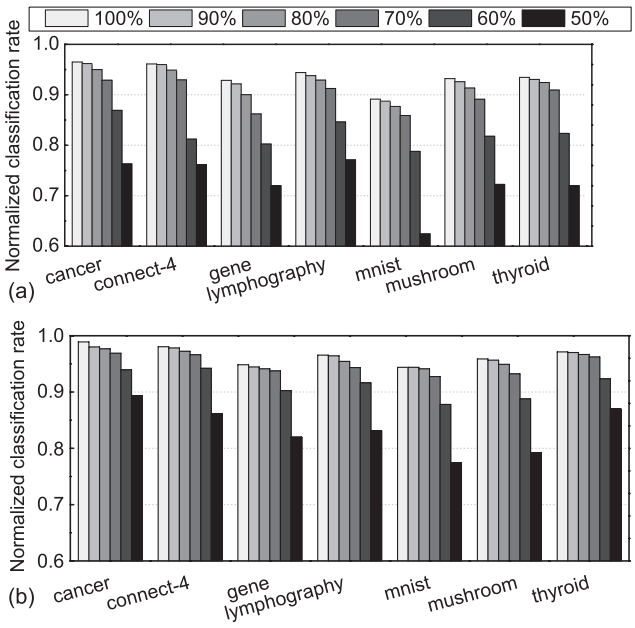


Fig. 9. The normalized classification rates of (a) MLP and (b) AAM under different MBC training efforts. The DAC/ADC resolution is set to 4-bit.

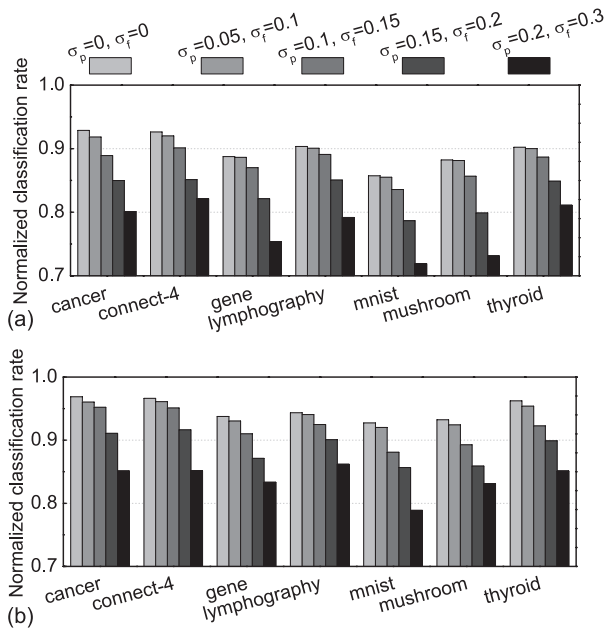


Fig. 10. The impact of device variations and signal fluctuations on computation accuracy: (a) MLP, (b) AAM.

the training effort reduces due to the degraded training accuracy. Compared to AAM which benefiting from the iterative computation, the classification rate of MLP decreases more obliviously when reducing the training effort. In particular, the classification rate reduces more prominently in the benchmarks that have a large scale network, such as *gene*, *mnist* and *mushroom*. When the training effort is set to 70%, the normalized classification rate is maintained above 86% for all benchmarks. Further reducing the size of training data set will quickly deteriorate the NCA computation accuracy. For the same reason, AAM shows a high tolerance to the degradation of training effort. In the following experiments, we adopt the training effort of 70% which provide a sufficient accuracy in both MLP and AAM implementations with negligible performance and energy overheads.

B. Impact of Device Variations and Signal Fluctuations

Fig. 10 shows the impacts of device variations and signal fluctuations on the computation accuracy of NCAs. Here σ_p represents the deviation of memristor resistance introduced by process variations. σ_f is the deviation of the analog signal magnitude, which are generated in DA/AD conversion, routing/buffering, sum-amplifier, and sigmoid function. As pointed out in [20], σ_f has greater impact on the computation accuracy of MBCs compared to σ_p. To be conservative, we use very pessimistic settings of σ_f to cover even the extreme cases in all simulations.

As the variations increase, the classification rate of the NCA generally degrades as expected though each benchmarks react to the change of the variations differently. Interestingly, the computation accuracy of *mnist* degrades slightly faster than other benchmarks, indicating a less robust ANN topology. Nonetheless, both MLP and AAM can maintain an acceptable computation accuracy when σ_p and σ_f are within a realistic range, i.e., $\sigma_p = 0.05$ and $\sigma_f = 0.1$. Again, AAM implementation demonstrates better reliability than MLP. After this section, all the simulations are performed at a nominal condition. However, by following the same flow that generates Fig. 10, the relevant statistical analysis can be easily conducted.

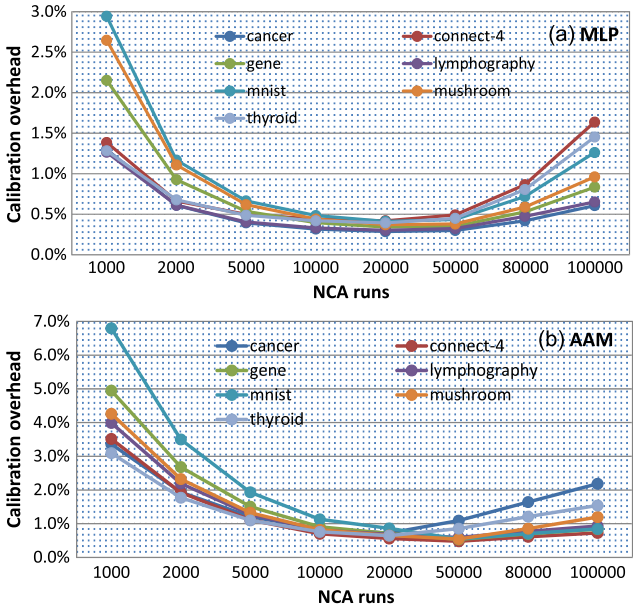


Fig. 11. The calibration overhead of (a) MLP and (b) AAM.

C. Design Tradeoff of Inline Calibration

We define the *calibration overhead* (OH_{cal}) to measure the impact of the inline calibration scheme on NCA performance within a calibrated execution period as

$$OH_{\text{cal}} = \frac{T_{\text{cal}}}{T_{\text{itvl}} + T_{\text{cal}}}. \tag{2}$$

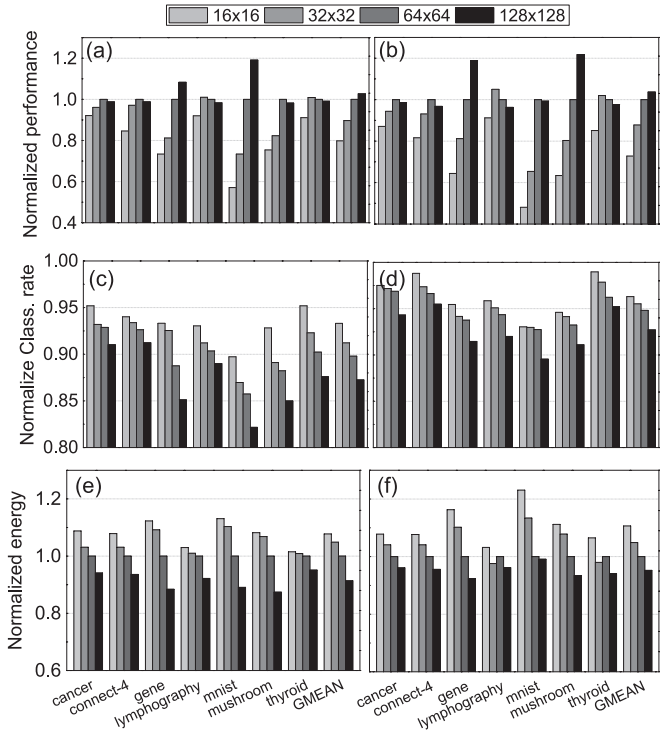


Fig. 12. The normalized NCA performance and energy at different MBC sizes in (a), (e) MLP and (b), (f) AAM implementations. The results of 64×64 MBC is used as normalization baseline. The classification rate at different MBC sizes in (c) MLP and (d) AAM. Results are normalized to the ideal case defined in Section VI-A.

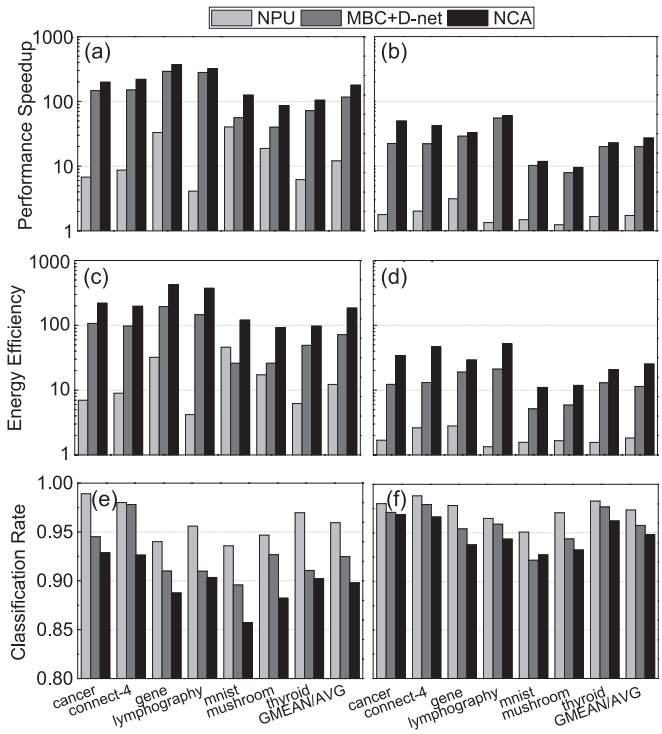


Fig. 13. The performance speedup, energy efficiency and classification rate of three ANN accelerator designs with MLP (a), (c), (e), and AAM (b), (d), (f) implementations.

Here T_{itvl} represents the execution time period of the NCA between two calibrations. T_{cal} is the time spent on the NCA calibration. As we discussed in Section IV-C, a longer T_{itvl} does not necessarily produce a smaller overhead.

Fig. 11 summarizes the OH_{cal} of all benchmarks with two ANN implementations when the T_{itvl} varies from 1000 to 100 000 NCA runs. The inline calibration is conducted at the end of every T_{itvl} to ensure the normalized classification rate of the NCA above a predetermined threshold, i.e., 84% for MLP and 90% for AAM. In both ANN implementations, the minimum OH_{cal} is achieved between 10 000 $\sim$ 50 000 NCA runs, depending on specific applications. The calibration time of AAM is longer than that of MLP due to its severer memristor resistance drift after the same number of NCA runs. In the following simulations, we choose a T_{itvl} of 20 000 NCA runs, which results in negligible calibration overhead in both ANN implementations (i.e., $< 0.41\%$ in MLP and $< 0.86\%$ in AAM, respectively).

D. Impact of MBC Sizes

A larger MBC array not only promotes the computation efficiency of the NCA by running more calculations at the same time but also decreases the latency and energy overhead of routing signals among MBC arrays when the scale of the ANN is too large to fit in one MBC array. However, the computation accuracy of a larger MBC array may be decreased due to process variations and signal fluctuations.

Fig. 12 shows the experiment results of the execution time, the classification rate and the energy consumption of Harmonica with different MBC sizes. As shown in Fig. 12(a), (b), following the increase of the MBC size, the system performance keeps improving as long as the size of the ANN is larger than the MBC. Further enlarging MBC, however, does not benefit the NCA performance. Although the energy consumption is reduced when the MBC size decreases due to the less data routing among M-Net [see Fig. 12(e), (f)], the computation accuracy keeps dropping down due to process variations and signal fluctuations [see Fig. 12(c), (d)]. Hence, we chose 64×64 MBCs as the default design in our simulations to balance between NCA computation efficiency and accuracy for the selected benchmarks.

E. Comparison to Other Design Alternatives

We evaluate the performance, energy efficiency, and classification rate of the three ANN accelerator designs: D-NPU, MBC+D-Net, and NCA, in the seven selected benchmarks, as shown in Fig. 13. The performance and energy efficiency are normalized to the baseline CPU execution, that is, running the MLP/AAM implementation exclusively on the CPU with FANN library. The results show that all the three ANN accelerators dramatically speedup the execution of the selected ANN benchmarks with slight degradation in computation accuracy w.r.t. the baseline CPU.

As shown in Fig. 13(a) and (b), the *geometric mean speedup* (GMS) achieved by D-NPU in digital format is $11.9\times$ or $1.7\times$ for MLP or AAM implementation, respectively. As a PE can process only one multiply-add operation per cycle, the computation bandwidth of D-NPU is relatively limited compared to the other two designs. MBCs+D-Nets utilizes MBC arrays for analog computation, which dramatically boosts the GMS of its MLP and AAM implementations to $117.2\times$ and $20.1\times$, respectively. Compared with MBCs+D-Nets, the proposed NCA

minimizes the costly DA/AD conversions and hence, demonstrates even higher speedup: The corresponding GMS values further rise to 178.41× (MLP) and 27.06× (AAM). In general, MLP achieves much faster execution than AAM because all the inputs traverse the network only once.

The energy efficiency result of each design is shown in Fig. 13(c) and (d), which follow the trend similar to the corresponding performance speedup. Compared to the baseline CPU architecture, 184.24× and 25.23× average energy savings are achieved by MLP and AAM, respectively. The energy efficiency of NCA is more than 2× higher than that of MBC+D-Net due to the dramatically reduced DA/AD conversion overhead. Note that in MLP, the energy efficiency of D-NPU under *MNIST* is higher than that of MBCs+D-Net. It is because the partitioning of *MNIST* into multiple MBCs introduces considerably large amount of data traffic among the MBCs and hence, significantly raises the AD/DA energy consumption in MBCs+D-Net.

Fig. 13(e) and (f) illustrate the classification rates obtained from each designs. In MLP, the accuracy of the NCA is the lowest among all the designs. The computation accuracy is enhanced in MBC+D-Net by utilizing digital network for signal transmission. As expected, the full digital implementation of D-NPU achieves the highest classification rate. In AAM, the three designs all achieve very high (i.e., 92%) and close accuracy in all benchmarks, say, with a variation less than 2.8%. This is because AAM can automatically compensate the adverse impact of the less reliable executions in each loop on the computation accuracy. As shown in Fig. 13(a) and (b), compared to MBC+D-Net, the performance speedup achieved by NCA in AAM implementation is only 1.3×, which is less than the 1.5× speedup achieved in MLP implementation.

In short, NCA exhibits extremely high performance and power efficiency while well maintains the computation accuracy within an acceptable level. Moreover, MLP and AAM implementations present different tradeoffs between the computation efficiency and accuracy, offering valuable design flexibility adaptive to the nature of the particular applications.

VII. Related Works

Artificial neural network (ANN) is recently gaining considerable attentions in computer architecture and solid state circuit societies [51], [52] as a promising candidate to conquer the well-known von Neumann bottleneck. Many studies have been conducted on the programming models for the ANN-based computing platform [53] and neuromorphic computing [54], which is referred to the VLSI implementation of ANN. To bridge the gap between ANN algorithm and its realization on microarchitectures, Hashmi *et al.* [53] proposed a neuromorphic *instruction set architecture* (ISA) to extract the representation of ANN and designed a runtime environment to generate the neuromorphic codes for different computing platforms, e.g., CPU, GPU, and Boolean logics. In this work, we leverage the advanced memristor crossbar technology to build Harmonica—a mixed-signal heterogeneous computing framework which offers much higher computing and power efficiency in ANN and learning applications than the system with conventional CMOS-only ANN accelerators.

Harmonica is able to perform not only ANN computations but also approximate computations, similar to the proposal in [13], [55]. Harmonica can be dynamically reconfigured to construct different ANN topologies, outperforming other digital ANN accelerators with time-multiplexed computation structure. Also, compared to conventional mixed-signal designs [56] with CMOS-based computing components, Harmonica demonstrates very attractive performance/power/area efficiency by utilizing the novel designs of memristor-based computing and analog routing scheme. Meanwhile, the computation accuracy of Harmonica is well maintained by our robust designs (e.g., M-Net) and inline calibration scheme, as demonstrated in the experiments.

VIII. Conclusion and Future Works

In this work, we propose a heterogeneous computing framework named *Harmonica*, which contains a memristor-based neuromorphic computing accelerator (NCA) tightly coupled with the general-purpose pipeline. Harmonica accomplishes 177.67× (27.2×) performance speedup and 184.71× (25.18×) energy reduction on average in the seven simulated benchmarks in MLP (AAM) implementations. The high computation and energy efficiency mainly come from: 1) the high-throughput of the mixed-signal NCA computation; 2) the excellent reconfigurability of the hierarchical memristor crossbar array structure; 3) the low data transmission overhead on the mixed-signal interconnection network (M-Net); and 4) the concise coordination interface between the general-purpose pipeline and the NCA. An inline calibration scheme is also developed to control the run-time NCA computation accuracy degradation incurred by memristor resistance shift. Besides the presented MLP and AAM networks, the NCA structure can construct other types of ANN by reconfiguring the M-Net and properly migrating data among the MBC arrays.

References

[1] A. Lukefahr, S. Padmanabha, R. Das, F. M. Sleiman, R. G. Dreslinski, T. F. Wenisch, and S. A. Mahlke, "Composite cores: Pushing heterogeneity into a core," in *Proc. MICRO*, 2012, pp. 317–328.

[2] K. Fan, M. Kudlur, G. Dasika, and S. Mahlke, "Bridging the computation gap between programmable processors and hardwired accelerators," in *Proc. HPCA*, 2009, pp. 313–322.

[3] A. R. Putnam, D. Bennett, E. Dellinger, J. Mason, and P. Sundararajan, "Chimps: A high-level compilation flow for hybrid cpu-fpga architectures," in *Proc. FPGA*, 2008, pp. 261–261.

[4] F. Song, S. Tomov, and J. Dongarra, "Enabling and scaling matrix computations on heterogeneous multi-core and multi-gpu systems," in *Proc. Supercomputing*, 2012, pp. 365–376.

[5] J. Sampson, G. Venkatesh, N. Goulding-Hotta, S. Garcia, S. Swanson, and M. B. Taylor, "Efficient complex operators for irregular codes," in *HPCA*, 2011, pp. 491–502.

[6] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, S. Swanson, and M. B. Taylor, "Conservation cores: Reducing the energy of mature computations," in *Proc. ASPLOS*, 2010, pp. 205–218.

[7] V. Govindaraju, C.-H. Ho, and K. Sankaralingam, "Dynamically specialized datapaths for energy efficient computing," in *Proc. HPCA*, 2011, pp. 503–514.

[8] S. Gupta, S. Feng, A. Ansari, S. A. Mahlke, and D. I. August, "Bundled execution of recurring traces for energy-efficient general purpose processing," in *Proc. MICRO*, 2011, pp. 12–23.

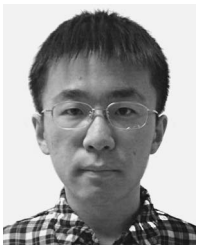
[9] T. Chen, Y. Chen, M. Duranton, Q. Guo, A. Hashmi, M. H. Lipasti, A. Nere, S. Qiu, M. Sebag, and O. Temam, "Benchnn: On the broad potential application scope of hardware neural network accelerators," in *Proc. IISWC*, 2012, pp. 36–45.

[10] B. Belhadj, A. Joubert, Z. Li, R. Héliot, and O. Temam, "Continuous real-world inputs can open up alternative accelerator designs," in *Proc. ISCA*, 2013, pp. 1–12.
[11] O. Temam, "A defect-tolerant accelerator for emerging high-performance applications," in *ISCA*, 2012, pp. 356–367.
[12] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, "Nanoscale memristor device as synapse in neuromorphic systems," *Nano Lett.*, vol. 10, no. 4, pp. 1297–1301, 2010.
[13] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural acceleration for general-purpose approximate programs," in *Proc. MICRO*, 2012, pp. 449–460.
[14] G. Indiveri, B. Linares-Barranco, R. Legenstein, G. Deligeorgis, and T. Prodromakis, "Integration of nanoscale memristor synapses in neuromorphic computing architectures," *Nanotechnology*, vol. 24, no. 38, 2013, Art no. 384010.
[15] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, pp. 80–83, 2008.
[16] A. S. Cassidy, P. Merolla, J. V. Arthur, S. Esser, B. Jackson, R. Alvarez-Icaza, P. Datta, J. Sawada, T. M. Wong, V. Feldman, A. Amir, D. B. dayan Rubin, E. Mcquinn, W. P. Risk, and D. S. Modha, "Cognitive computing building block: A versatile and efficient digital neuron model for neurosynaptic cores," in *Proc. International Joint Conference on Neural Networks (IJCNN). IEEE*, 2013, pp. 1–10.
[17] D. Kadetotad, Z. Xu, A. Mohanty, P.-Y. Chen, B. Lin, J. Ye, S. Vrudhula, S. Yu, Y. Cao, and J.-S. Seo, "Parallel architecture with resistive crosspoint array for dictionary learning acceleration," *Circuits Syst.*, vol. 5, no. 2, pp. 194–204, 2015.
[18] S. O. Haykin, *Neural Networks and Learning Machines*. London, U.K.: Prentice-Hall, 2008.
[19] H. Wang, Y. Wu, B. Zhang, and K. L. Du, "Recurrent neural networks: Associative memory and optimization," *J Inform Tech Soft Engg*, 2011.
[20] B. Liu, M. Hu, H. Li, and Y. chen, "Digital assisted noise eliminating training for memristor crossbar based analog neuromorphic computing engine," in *Proc. DAC*, 2013.
[21] L. O. Chua, "Memristor-the missing circuit element," *IEEE Trans. Circuit Theory*, vol. 18, no. 5, pp. 507–519, 1971.
[22] S. Shin, K. Kim, and S.-M. Kang, "Memristor applications for programmable analog ics," *IEEE Trans. Nanotechnol.*, vol. 10, no. 2, pp. 266–274, 2011.
[23] L. Zhang, N. Ge, J. J. Yang, Z. Li, R. S. Williams, and Y. Chen, "Low voltage two-state-variable memristor model of vacancy-drift resistive switches," *Appl. Phys. A*, vol. 119, no. 1, pp. 1–9, 2015.
[24] F. Bedeschi, R. Fackenthal, C. Resta, E. M. Donzé, M. Jagasivamani, E. C. Buda, F. Pellizzer, D. W. Chow, A. Cabrini, and G. M. A. Calvi, *et al.*, "A bipolar-selected phase change memory featuring multi-level cell storage," *Solid-State Circuits*, vol. 44, no. 1, pp. 217–227, 2009.
[25] W. Xu, Y. Chen, X. Wang, and T. Zhang, "Improving stt mram storage density through smaller-than-worst-case transistor sizing," in *Proc. DAC*, 2009, pp. 87–90.
[26] K.-H. Kim, S. Gaba, D. Wheeler, J. M. Cruz-Albrecht, T. Hussain, N. Srinivasa, and W. Lu, "A functional hybrid memristor crossbar-array/cmos system for data storage and neuromorphic applications," *Nano Lett.*, vol. 12, no. 1, pp. 389–395, 2011.
[27] F. Alibart, L. Gao, B. D. Hoskins, and D. B. Strukov, "High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," *Nanotechnology*, vol. 23, no. 7, 2012.
[28] L. O. Chua and S.-M. Kang, "Memristive devices and systems," *Proc. IEEE*, vol. 64, no. 2, pp. 209–223, 1976.
[29] P.-Y. Chen, D. Kadetotad, Z. Xu, A. Mohanty, B. Lin, J. Ye, S. Vrudhula, J.-S. Seo, Y. Cao, and S. Yu, "Technology-design co-optimization of resistive cross-point array for accelerating learning algorithms on chip," in *Proc. Design, Autom. Test Eur. Conf. Exhi.*, 2015, pp. 854–859.
[30] M. Hu, H. Li, Q. Wu, and G. S. Rose, "Hardware realization of bsb recall function using memristor crossbar arrays," in *Proc. DAC*, 2012, pp. 498–503.
[31] A. Joubert, B. Belhadj, O. Temam, and R. Heliot, "Hardware spiking neurons design: Analog or digital?" in *Proc. Int. Joint Conf. Neural Netw.*, 2010, pp. 1–5.
[32] S. Yu, Y. Wu, and H.-S. P. Wong, "Investigating the switching dynamics and multilevel capability of bipolar metal oxide resistive switching memory," *Appl. Phys. Lett.*, vol. 98, no. 10, pp. 103 514–103 514-3, 2011.
[33] J. Balfour and W. J. Dally, "Design tradeoffs for tiled cmp onchip networks," in *Proc. ICS*, 2006, pp. 187–198.
[34] H. Li, B. Liu, X. Liu, M. Mao, Y. Chen, Q. Wu, and Q. Qiu, "The applications of memristor devices in next-generation cortical processor designs," in *Proc. ISCAS*, 2015, pp. 17–20.
[35] L. Dai and R. Harjani, "Cmos switched-op-amp-based sample-and-hold circuit," in *Proc. Solid-State Circuits*, 2000.
[36] L. Gao, P.-Y. Chen, and S. Yu, "Programming protocol optimization for analog weight tuning in resistive memories," *IEEE Electron Device Lett.*, vol. 36, no. 11, pp. 1157–1159, 2015.
[37] B. J. Choi, A. B. Chen, X. Yang, and I.-W. Chen, "Purely electronic switching with high uniformity, resistance tunability, good retention in pt-dispersed sio2 thin films for reram," *Adv. Mater.*, vol. 23, no. 33, pp. 3847–3852, 2011.
[38] W. Zhao and Y. Cao, "Predictive technology model for nano-cmos design exploration," in *JETC*, 2007, vol. 3, no. 1, pp. 1.
[39] M. Gustavsson, J. J. Wikner, and N. Tan, CMOS Data Converters for Communications, 2000.
[40] F. Krummenacher, "Micropower switched capacitor biquadratic cell," *Solid-State Circuits*, vol. 17, no. 3, pp. 507–512, 1982.
[41] Virtuoso. [Online]. Available: http://www.cadence.com/products/cic/pages/default.aspx
[42] S. Shapero and P. Hasler, "Mismatch characterization and calibration for accurate and automated analog design," *Circuits Syst.*, vol. 60, no. 3, pp. 548–556, 2013.
[43] L. Prechelt, "Proben1-a Set of Neural Network Benchmark Problems and Benchmarking Rules," University of Karlsruhe Tech. Rep., 1994.
[44] Uci Machine Learning Repository. [Online]. Available: http://archive.ics.uci.edu/ml/
[45] The Mnist Database of Handwritten Digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[46] S. Nissen, "Implementation of a Fast Artificial Neural Network Library (fann)," Department of Computer Science, University of Copenhagen Tech. Rep., 2003.
[47] Macsim. [Online]. Available: http://code.google.com/p/macsim/
[48] Intel Atom Proccesor. [Online]. Available: http://ark.intel.com/products/family/29035/
[49] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "Mcpat: An integrated power, area, timing modeling framework for multicore and manycore architectures," in *Proc. MICRO*, 2009, pp. 469–480.
[50] N. Jian, D. U. Becker, G. Michelogiannakis, J. Balfour, B. Towles, J. Kim, and W. J. Dally, A Detailed and Flexible Cycle-Accurate Network-On-Chip Simulator, 2013.
[51] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proc. ICML*, 2007, pp. 473–480.
[52] O. Temam, The Rebirth of Neural Networks, Saint Malo, France, Jun. 2010. [Online]. Available: http://pages.saclay.inria.fr/olivier.temam/homepage/ISCA2010web.pdf
[53] A. Hashmi, A. Nere, J. J. Thomas, and M. Lipasti, "A case for neuromorphic isas," in *Proc. ASPLOS*, 2011, pp. 145–158.
[54] J. sun Seo, B. Brezzo, Y. Liu, B. D. Parker, S. K. Esser, R. K. Montoye, B. Rajendran, J. A. Tierno, L. Chang, D. S. Modha, and D. J. Friedman, "A 45 nm cmos neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," in *Proc. CICC*, 2011, pp. 1–4.
[55] X. Liu, M. Mao, H. Li, Y. Chen, H. Jiang, J. J. Yang, Q. Wu, and M. Barnell, "A heterogeneous computing system with memristor-based neuromorphic accelerators," in *Proc. HPEC*, 2014, pp. 1–6.
[56] R. S. Amant, A. Yazdanbakhsh, J. Park, B. Thwaites, H. Esmaeilzadeh, A. Hassibi, L. Ceze, and D. Burger, "General-purpose code acceleration with limited-precision analog computation," in *Proc. ISCA*, 2014.

Xiaoxiao Liu is a Ph.D. student at the Electrical and Computer Engineering department, University of Pittsburgh. She is working with Dr. Yiran Chen on emerging memory and heterogeneous system architecture. She received the BS and MS degrees in Electrical Engineering and Software Engineering from Beihang University, Beijing, China, in 2005 and 2009, respectively. Before Ph.D. study, she worked as a Senior Design/Layout engineer in AMD Shanghai Design Center and Digital IC Design Engineer in Japan.

Mengjie Mao is a Ph.D student in the Electrical and Computer Engineering department at University of Pittsburgh. His research interests are GPGPU, emerging memory technologies and neuromorphic computing.

Beiye Liu is currently a Ph.D. candidate at the Electrical and Computer Engineering department, University of Pittsburgh. He is working with Yiran Chen on brain-inspired emerging computing hardware architectures for machine learning algorithms. In 2013, he worked as visiting scholar at HP Research Lab and he also worked as an intern software engineer at Uber Advanced Technology Center in 2015. He authored/co-authored multiple papers published on top conferences and journals including Design Automation Conference (DAC), International Conference on Computer-Aided Design (ICCAD), Neural Processing Letters and etc.



Boxun Li received B.S. degree in Electronic Engineering from Tsinghua University, China, in 2013. He is currently pursuing his M.S. degree in Department of Electronic Engineering, Tsinghua University. His research mainly focuses on energy efficient hardware computing system design, and parallel computing based on GPU.

Yu Wang (S'05–M'07–SM'14) received his B.S. degree in 2002 and Ph.D. degree (with honor) in 2007 from Tsinghua University, Beijing, China. He is currently an Associate Professor with the Department of Electronic Engineering, Tsinghua University. His research interests include parallel circuit analysis, application specific hardware computing (especially on the Brain related problems), and power/reliability aware system design methodology. Dr. Wang has authored and coauthored over 140 papers in refereed journals and conferences. He is the recipient of IBM X10 Faculty Award in 2010, Best Paper Award in ISVLSI 2012, Best Poster Award in HEART 2012, and 6 Best Paper Nomination in ASPDAC/CODES/ISLPED.

Hao Jiang (M'00) received the B.S. degree in materials sciences from Tsinghua University, China, in 1994 and the Ph.D. degree in electrical engineering from the University of California, San Diego, in 2000. Hao Jiang has been with San Francisco State University since August 2007. Currently, he is an Associate Professor in electrical engineering. Prior joining SFSU, he worked for Broadcom Corporation, Jazz Semiconductor and Conexant Systems Inc. His research interests are in the general area of analog and radio-frequency integrated circuits and systems.

Mark Barnell (M'09) received the B.S. degree of Optical Engineering from University of Rochester in 1987 and the M.S. degree in Computer Science from SNUY Polytechnic Institute in 2000.

He is a Senior Computer Scientist with the U.S. Air Force Research Laboratory, high performance computing systems branch (AFRL/RITB). Mr. Barnell currently is the HPC Director for AFRLs Information Directorate Computing and Communications Affiliate Resource Center (ARC) and Agile High Performance Systems (AHPS) Program Manager. His areas of expertise include high performance computers, embedded computing, persistent wide area surveillance, distributed and next generation architectures.

Qing Wu (M'01) received the B.S. and M.S. degrees from the Department of Information Science and Electronic Engineering at Zhejiang University, Hangzhou, China, in 1993 and 1995, respectively, and the Ph.D. degree from the Department of Electrical Engineering at the University of Southern California in 2002. Currently, he is a Senior Electronics Engineer at the United States Air Force Research Laboratory (AFRL), Information Directorate (RI). Before joining AFRL, he was an Assistant Professor in the Department of Electrical and Computer Engineering at State University of New York, Binghamton. His research interests include large-scale neuromorphic computing circuits and systems, high-performance computing architectures, energy-efficient embedded computing.

Jianhua (Joshua) Yang (M'08) received the B.A. degree in mechanical engineering from Southeast University in China in 1997 and the Ph. D. degree in Material Science Program from the University of Wisconsin Madison in 2007. He is currently a Professor in the Department of Electrical and Computer Engineering department of the University of Massachusetts, Amherst. He spent over 8 years at HP Labs before joining UMass in 2015. His current research interests are Nanoelectronics and Nanoionics, especially for unconventional computing applications, where he authored and co-authored over 100 papers in peer-reviewed academic journals and conferences, and holds 61 granted and over 70 pending US Patents.

Hai (Helen) Li (M'08–SM'16) received the B.S. and M.S. degrees in microelectronics from Tsinghua University, Beijing, China, and the Ph.D. degree from the Electrical and Computer Engineering Department at Purdue University, West Lafayette, IN, USA, in 2004. She is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Pittsburgh. Prior to it, she was with Qualcomm Inc., Intel Corp., Seagate Technology, and Polytechnic Institute of New York University. Her research interests include memory design and architecture, neuromorphic architecture for brain-inspired computing systems, architecture/circuit/device cross-layer optimization for low power and high performance. She has 100+technical papers published in refereed journals and conferences and 61 U.S. patents granted.

Yiran Chen (M'04–SM'16) received B.S and M.S. (both with honor) from Tsinghua University and Ph.D. from Purdue University in 2005. After five years in industry, he joined University of Pittsburgh in 2010 as Assistant Professor and then promoted to Associate Professor in 2014. He is now holding Bicentennial Alumni Faculty Fellow and co-directing Evolutionary Intelligence Lab (www.ei-lab.org) at Electrical and Computer Engineering Department, focusing on the research of nonvolatile memory and storage systems, neuromorphic computing, and mobile systems. Dr. Chen has published one book, a handful of book chapters, and more than 200 journal and conference papers. He has been granted with 89 US and international patents with other 13 pending applications.