Neural Network-Inspired Analog-to-Digital Conversion to Achieve Super-Resolution with Low-Precision RRAM Devices

Weidong Cao*, Liu Ke*, Ayan Chakrabarti** and Xuan Zhang*
* Department of ESE, ** Department of CSE, Washington University, St.louis, MO, USA

***Abstract*— Recent works propose neural network- (NN-) inspired analog-to-digital converters (NNADCs) and demonstrate their great potentials in many emerging applications. These NNADCs often rely on resistive random-access memory (RRAM) devices to realize the NN operations and require high-precision RRAM cells (6∼12-bit) to achieve a moderate quantization resolution (4∼8-bit). Such optimistic assumption of RRAM resolution, however, is not supported by fabrication data of RRAM arrays in large-scale production process. In this paper, we propose an NN-inspired super-resolution ADC based on low-precision RRAM devices by taking the advantage of a co-design methodology that combines a pipelined hardware architecture with a custom NN training framework. Results obtained from SPICE simulations demonstrate that our method leads to robust design of a 14-bit super-resolution ADC using 3-bit RRAM devices with improved power and speed performance and competitive figure-of-merits (FoMs). In addition to the linear uniform quantization, the proposed ADC can also support configurable high-resolution nonlinear quantization with high conversion speed and low conversion energy, enabling future intelligent analog-to-information interfaces for near-sensor analytics and processing.**

I. Introduction

Many emerging applications have posed new challenges for the design of conventional analog-to-digital (A/D) converters (ADCs) [1]–[4]. For example, multi-sensor systems desire programmable nonlinear A/D quantization to maximize the extraction of useful features from the raw analog signal, instead of directly performing uniform quantization by conventional ADCs [3], [4]. This can alleviate the computational burden and reduce the power consumption of back-end digital processing, which is the dominant bottleneck in intelligent multi-sensor systems. However, such flexible and configurable quantization schemes are not readily supported by conventional ADCs with dedicated circuitry that has fixed conversion references and thresholds.

To overcome this inherent limitation of conventional ADCs, several recent works [5]–[7] have introduced neural network-inspired ADCs (NNADCs) as a novel approach to designing intelligent and flexible A/D interfaces. For instance, a learnable 8-bit NNADC [7] is presented to approximate multiple quantization schemes where the NN weight parameters are trained off-line and can be configured by programming the same hardware substrate. Another example is a 4-bit neuromorphic ADC [6] proposed for general-purpose data conversion using on-line training by leveraging the input amplitude statistics and application sensitivity. These NNADCs are often built on resistive random-access memory (RRAM) crossbar array to realize the basic NN operations, and can be trained to approximate the specific quantization/conversion functions required by different systems. However, a major challenge for designing such NNADCs is the limited conductance/resistance resolution of RRAM devices. Although these NNADCs optimistically assume that each RRAM cell can be precisely programmed with 6∼12-bit resolution, measured data from realistic fabrication process suggest the actual RRAM resolution tends to be much lower (2∼4-bit) [8], [9]. Therefore, there exists a gap between the reality and the assumption of RRAM precision, yet lacks a design methodology to build super-resolution NNADCs from low-precision RRAM devices.

In this paper, we bridge this gap by introducing an NN-inspired design methodology that constructs super-resolution ADCs with low-precision RRAM devices. Taking advantage of a co-design methodology that combines a pipelined hardware architecture with deep learning-based custom training framework, our method is able to achieve an NN-inspired ADC whose resolution far exceeds the precision of the underlying RRAM devices. The key idea of a pipelined architecture is that many consecutive low-resolution (1∼3-bit) quantization stages can be cascaded in a chain structure to obtain higher resolution. Since each stage now only needs to resolve 1∼3-bit, we can accurately train and instantiate it with low-precision RRAM devices to approximate the ideal quantization functions and residue functions. Key innovations and contributions in this paper are as follow:

- We propose a co-design methodology leveraging pipelined hardware architecture and custom training framework to achieve super-resolution analog-to-digital conversion that far exceeds the limited precision of the RRAM device.
- We systematically evaluate the impacts of NN size and RRAM precision on the accuracy of NN-inspired sub-ADC and residue block and perform design space exploration to search for optimal pipelined stage configuration with balanced trade-off between speed, area, and power consumption.
- SPICE simulation results demonstrate that our proposed method is able to generate robust design of a 14-bit super-resolution NNADC using 3-bit RRAM devices. Comparisons with both the state-of-the-art ADCs and other NNADC designs reveal improved performance and competitive figure-of-merits (FoMs).
- Our proposed ADC can also support configurable nonlinear quantization with high-resolution, high conversion speed, and low conversion energy.

978-1-7281-2350-9/19/$31.00 ©2019 IEEE

II. Preliminaries

A. RRAM Device, Crossbar Array and NN

1) RRAM device: A RRAM device is a passive two-terminal element with variable resistance and possesses many special advantages, such as small cell size ($4F^2$, F–the minimum feature size), excellent scalability ($<10nm$), faster read/write time ($<10ns$) and better endurance ($\sim 10^{10}$ cycles) than Flash devices [2], [10].

2) RRAM crossbar array: RRAM devices can be organized into various ultra-dense crossbar array architectures. Fig. 1(a) shows a passive crossbar array composed of two sub-arrays to realize bipolar weights without the use of power-hungry operational-amplifiers (op-amps) [7]. The relationship between the input voltage "vector" ($\vec{V}_{\text{in}}$) and output voltage "vector" ($\vec{V}_{\text{o}}$) can be expressed as $V_{\text{o},j} = \sum_k W_{k,j} \cdot V_{\text{in},k} + V_{\text{off},j}$. Here, k ($k \in \{1, 2, \ldots, H\}$) and j ($j \in \{1, 2, \ldots, M\}$) are the indices of input ports and output ports of the crossbar array. The weight $W_{k,j}$ can be represented by the subtraction of two conductances in upper (U) sub-array and lower (L) sub-array as

$$W_{k,j} = (g^U_{k,j} - g^L_{k,j})/\sum, \quad \sum = \sum\nolimits_k (g^U_{k,j} + g^L_{k,j}). \quad (1)$$

Therefore, the RRAM crossbar array is capable of performing analog vector-matrix multiplication (VMM) and the parameters of the matrix rely on the RRAM resistance states.

3) Artificial NN: With the RRAM crossbar array, an NN shown in Fig. 1(b) can be implemented on such hardware substrate. Generally, the NN processes the data by executing the following operations layer-wise [17]:

$$\vec{y}_{i+1} = f(W_{i,i+1} \cdot \vec{x}_i + \vec{b}_{i+1}). \quad (2)$$

Here, $W_{i,i+1}$ is the weight matrix to connect the layer i and layer $(i+1)$. $f(\cdot)$ is a nonlinear activation function (NAF). These basic NN operations, e.g., VMM and NAF, can be mapped to the RRAM crossbar array and CMOS inverters shown in Fig. 1(a), where the voltage transfer characteristic (VTC) is used as an NAF [7].

B. NN-Inspired ADCs

ADC can be viewed as a special case of classification problems which maps a continuous analog signal to a multi-bit digital code. An NN can be trained to learn this input-output relationship, and a hardware implementation of this NN can be instantiated in the analog and mixed-signal domain. This is the basic idea behind NNADCs which implements the learned NN on a hardware substrate to approximate the desired quantization functions for data conversion:

$$\sum_{i=1}^{M} 2^{i-1} \cdot b_{\text{o},i} = \text{round}\left(\frac{V_{\text{in}} - V_{\min}}{V_{\max} - V_{\min}} \times (2^M - 1)\right), \quad (3)$$

where, M is the resolution; V_{in} is input analog signal and b_{o} is the output digital codes; $V_{\min}$ and $V_{\max}$ are the minimum and maximum values of the scalar input signal V_{in}. Since RRAM crossbar array provides a promising hardware substrate to build NNs, recent work has demonstrated several NNADCs based on RRAM devices [5]–[7]. Although the NN architectures adopted by these NNADCs are various, they all rely on a training process to learn the appropriate NN weights to approximate flexible quantization schemes that can be configured by programming the weights stored in RRAM conductance/resistance. However, existing NNADCs [5]–[7] often exhibit modest conversion resolution (4∼8-bit) and invariably rely on optimistic assumption of the RRAM precision (6∼12-bit), which is not well substantiated by measurement data from realistic RRAM fabrication process [8], [9]. This resolution limitation severely constrains the application of NNADCs in emerging multi-sensor systems that require high-resolution (>10-bit) A/D interfaces for feature extraction and near-sensor processing [1], [3], [4].

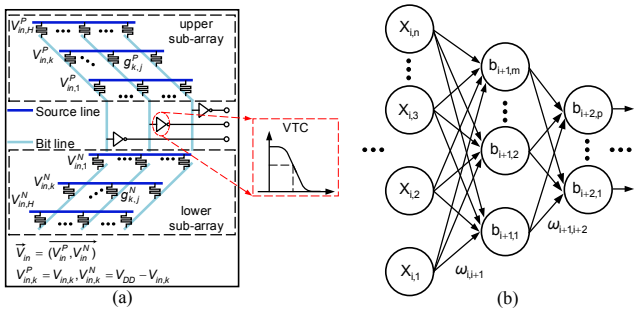

Fig. 1: (a) Hardware substrate to perform basic NN operations, where the passive crossbar array with two sub-arrays executes VMM and the VTC of CMOS inverter acts as NAF. (b) An example of NN.

C. Pipelined ADCs

Pipelined architecture is a well-established ADC topology to achieve high sampling rate and high resolution with low-resolution quantization stages [11]. Fig. 2(a) illustrates a typical pipelined ADC with M stages whose resolution RESO can be achieved by concatenating N_i-bit of each stage with digital combiner: RESO $= \sum_{i=1}^{M} N_i$. Note that N_i is usually ≤ 4 and not necessarily identical in all stages. As the Fig. 2(a) illustrates, an arbitrary stage-i contains two sub-blocks: a sub-ADC and a residue. The sub-ADC resolves N_i-bit binary codes D_{N_i} from input residue r_{i-1}, while the residue part amplifies the subtraction between the input residue r_{i-1} and the analog output of sub-ADC by 2^{N_i} to generate the output residue r_i for next stage. This process can be expressed as a simple function:

$$r_i = [r_{i-1} - \mathit{REF}(D_{N_i})] \cdot 2^{N_i}. \quad (4)$$

Here, $\mathit{REF}(D_{N_i})$ is the analog output of sub-DAC that depends on D_{N_i}. For example, assuming $r_{i-1} \in [0, V_{\text{DD}}]$ and $N_i = 1$, then $\mathit{REF}(0) = 0$ and $\mathit{REF}(1) = V_{\text{DD}}/2$; and Fig. 2(b) shows the corresponding residue function. To understand the basic working principle of pipelined ADCs, we use a 4-bit pipelined ADC with four 1-bit stages in Fig. 2(c) as an example. Assuming the initial analog input is $0.7V$ ($V_{\text{DD}} = 1V$), then the first stage will output "1"— a digital code, and "$0.4V$"— an analog residue according to Eq. (4) which will be processed by the following stage in the same way as initial analog input. Finally, we can obtain 4-bit outputs 1011, which is the quantization of $0.7V$ ($0.7/1 = 11.2/2^4 \approx 11/2^4$). This example also shows that a higher resolution (4-bit) can indeed be constructed with low-precision (1-bit) stages in a pipelined ADC.

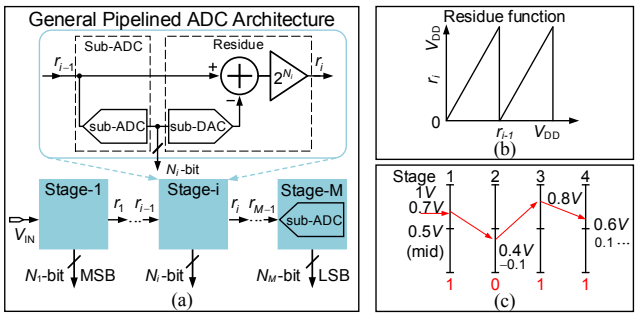

Fig. 2: (a) General architecture of pipelined ADC. (b) An example of residue function when $N_i = 1$. (c) A quantization example of a 4-bit pipelined ADC with four 1-bit stages.

III. Co-Design Methodology

A. Hardware Substrate

1) Pipelined architecture: The observation from traditional pipelined ADCs motivates us to extend such architecture to NNADC to enhance its resolution beyond the limit of RRAM precision. The overall hardware architecture for the proposed high-resolution NNADC is presented in Fig. 3(a), where a pipelined architecture composed of cascaded conversion stages is adopted in the design. This pipelined architecture brings two direct benefits. First, each stage in the proposed NNADC now only needs to resolve 1∼3-bit quantization, which is well within the precision limit of current RRAM fabrication process [8], [9] and can be easily achieved with the automated design methodology introduced in previous work [7]. Second, although many cascading stages are needed, there only exist three distinct low-resolution configurations to choose from for each stage, namely $N_i = 1, 2, 3$. This allows us to simplify the design process by focusing on optimizing the sub-block design of each stage with different resolutions. The full pipelined system can then be assembled by iterating through different combinations of the sub-blocks with different resolutions.

2) Low-resolution NNADC stage: For stage-i in the proposed NNADC, we use a five-layer NN to implement the sub-ADC and the residue block. The five-layer NN can be decomposed into two three-layer sub-blocks, and each of them can be mapped into the corresponding sub-ADC and residue in Fig. 2(a). The cornerstone of this mapping methodology is the universal approximation theorem that a feed-forward three-layer NN with a single hidden layer can approximate arbitrary complex functions [13]. We use the RRAM crossbar array and CMOS inverter illustrated in Fig. 1(a) as the hardware substrate to design the sub-blocks of each stage. As Fig. 3(b) shows, for the sub-ADC, the input analog signal represents the single "place holder" neuron in MLP's input layer. Therefore, the weight matrix dimensions are $H_{F,i} \times 1$ between the hidden and the input layer, and $H_{F,i} \times S_i$ between the hidden and the output layer, assuming there are $H_{F,i}$ and S_i neurons in the hidden and output layer. Here, we use a redundant "smooth" $S_i \rightarrow N_i$ encoding method to replace the standard N_i-bit binary encoding with S_i bits ($S_i > N_i$) according to previous work [7], as it improves the training accuracy and reduces hidden layer size of the sub-ADC. For example, we use $3 \rightarrow 2$ smooth codes to train a 2-bit sub-ADC with 3-bit smooth codes as output in Fig. 4(b). For the residue, there are $(1 + S_i)$ input neurons (one analog input and S_i-bit smooth digital codes from the proceeding sub-ADC block), and only one analog output neuron; therefore, the weight matrix dimensions are $H_{R,i} \times (1 + S_i)$ between the hidden and the input layer and $H_{R,i} \times 1$ between the hidden and the output layer, assuming there are $H_{R,i}$ hidden neurons. The sampling/hold (S/H) circuits [18] are used in the output layer to drive the next stage. Since the op-amps in Fig. 2(a) are eliminated in the NN-inspired design of residue circuit, considerable power saving can be obtained from each stage.

B. Training Framework

1) Training overview: We propose a training framework that accurately captures the circuit-level behavior of the hardware substrate in its mathematical model and is able to learn the robust NNs and its associated hardware design parameters (i.e., RRAM conductance) to approximate the sub-ADC and residue for each stage. The training framework incorporates two important features. First, we employ collaborative training for the two sub-blocks in each stage. The sub-ADC is initially trained to approximate the ideal quantization function with high-fidelity, then its digital outputs and original analog input are directly fed to the residue block for the residue training. This collaborative training flow can effectively minimize the discrepancy between the circuit artifacts and the ideal conversion at each stage. Second, non-idealities of devices, such as process, voltage and temperature (PVT) variations of the CMOS device and limited precision of the RRAM devices, can be incorporated into training to make the proposed NNADC robust to these defects [14]. This is another advantage of the proposed NNADC over traditional ADC designs, where even with delicate calibration techniques, the non-idealities cannot be fully mitigated [11].

2) Training steps: The detailed training flow is shown in Fig. 3(b), which consists of four steps. We focus on describing the training steps for the residue block, as we adopt similar sub-ADC training method that has been elaborated in previous work [7], [14].

Step ①: establish learning objective. For the residue circuit, its output is an analog value; therefore, the hardware substrate can be modeled as a three-layer NN with a "placeholder" output neuron:

$$\tilde{h}_i = L_1(r_{i-1}, D_{S_i}; \theta_{1,i}), \quad r_i = L_2(h_i; \theta_{2,i}). \tag{5}$$

Here, $h_i = \sigma_{\text{VTC,i}}(\tilde{h}_i)$. D_{S_i} indicates the digital output of the ADC ("1" means V_{DD}, and "0" means GND), and r_{i-1} is the scalar residue input of stage-i; $\tilde{h}_i$ denote the outputs of the first crossbar layer, which are modeled as a linear function L_1 of r_{i-1} and D_{S_i}, with learnable parameters $\theta_{1,i} = \{W_1, V_1\}$ corresponding to RRAM crossbar array conductances. Each of these voltages is passed through an inverter (shown in Fig. 1(a)), whose input-output relationship is modeled by the nonlinear function $\sigma_{\text{VTC,i}}(\cdot)$, to yield the vector h_i. The linear function L_2 models the second layer of the crossbar to

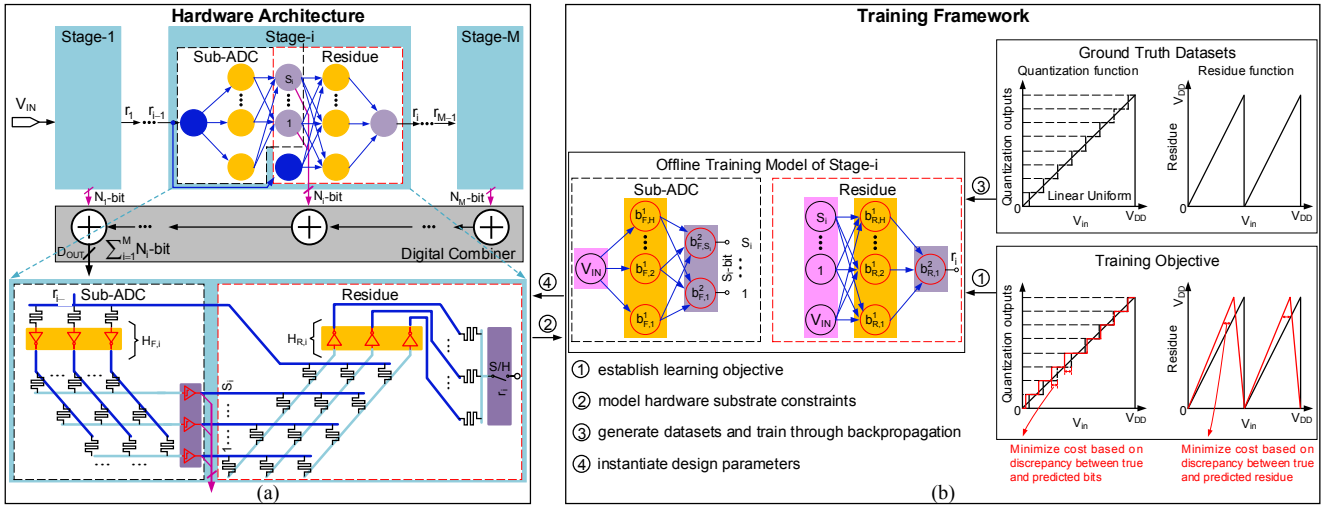

Fig. 3: Proposed co-design framework for the super-resolution NNADC. (a) Pipelined architecture for the proposed NNADC. (b) Off-line training model of each stage-i. Proposed training framework takes ground truth datasets as inputs during off-line training to find the optimal weights and derive the RRAM resistances to minimize cost function and best approximate ideal quantization function and residue function.

produce the output residue r_i for next stage, with learnable parameters $\theta_{2,i} = \{W_2, V_2\}$. The learning objective is to find optimal values for the parameters $\{\theta_{1,i}, \theta_{2,i}\}$ such that for all values of r_{i-1} in the input range, the circuit yields corresponding residue r_i that are equal or close to the desired "ground truth" r_{GT} in Eq. (4). To achieve this aim, we define a cost function $C(r_i, r_{\text{GT}})$ to measure the discrepancy between predicted r_i and true r_{GT} based on the mean-square loss:

$$C(r_i, r_{\text{GT}}) = \sum\nolimits_j [r_{\text{GT}}(j) - r_i(j)]^2. \tag{6}$$

Step ②: model hardware constraints. Hardware constraints come from three aspects: CMOS neuron PVT variations, limited precision of RRAM device, and passive crossbar array. To reflect these hardware constraints, we first group all VTCs obtained by Monte Carlo simulations in A_{VTC} using the technology specification in Section IV-A. Meanwhile, we control the precision of weight with A_R-bit during the training. Finally, we let the summation of all elements (absolute value) in each column ("0") of $W_{1,2}$ be < 1:

$$\sum(abs(W_1), 0) < 1; \quad \sum(abs(W_2), 0) < 1, \tag{7}$$

to reflect the weights constraints in Eq. (1).

Step ③: hardware-oriented training. We initialize the parameters $\{\theta_{1,i}, \theta_{2,i}\}$ randomly, and update them iteratively based on gradients computed on mini-batches of $\{(r_{i-1}, D_{N_i}, r_{\text{GT}})\}$ pairs randomly sampled from the input range. To incorporate the hardware constraints in step ② into training, we let each neuron j in Eq. (5) randomly pick up a VTC from A_{VTC} during training:

$$\sigma^j_{\text{VTC,i}} = A_{\text{VTC}}[\text{randint}(N)], j = 1, 2, ..., H_{R,i}. \tag{8}$$

We then periodically clip all values of W_1 between $[-1/(1+N_i), 1/(1+N_i)]$, as well as W_2 between $[-1/H_{R,i}, 1/H_{R,i}]$ to satisfy Eq. (7).

Step ④: instantiate conductance values. We adopt the same instantiation method based on previous work [7], which is proven to always find a set of equivalent conductances from the trained weights and biases to map to the RRAM devices in the hardware substrate. After this, we perturb each resistance R by:

$$R \leftarrow R \cdot e^{\theta}; \quad \theta \sim N(0, \sigma^2), \tag{9}$$

to evaluate the robustness of the NN model to the stochastic variation of RRAM resistance [2].

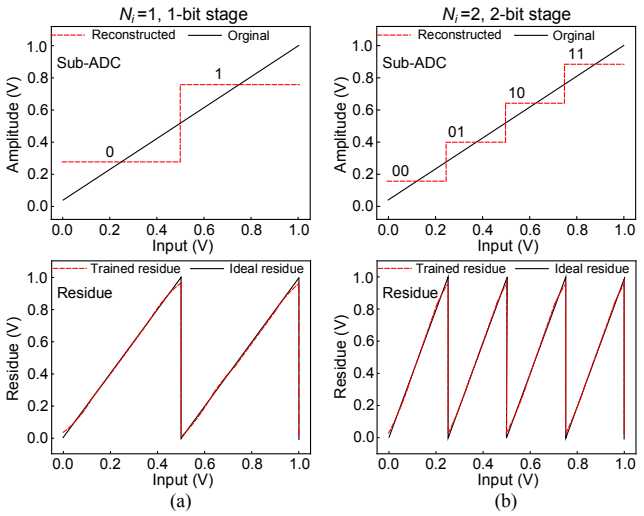

Fig. 4: Illustrations of trained sub-ADC and residue functions for a pipeline stage with different resolution. (a) 1-bit stage ($N_i = 1$). (b) 2-bit stage ($N_i = 2$).

C. Examples of Trained Sub-ADC and Residue

Fig. 4 illustrates the SPICE simulation of different trained stages with the proposed training framework. The sub-ADC and the residue in Fig. 4(a) are trained through a $1 \times 3 \times 2$ NN and a $3 \times 5 \times 1$ NN respectively by setting $N_i = 1$, while the sub-ADC and the residue in Fig. 4(b) are trained through a $1 \times 4 \times 3$ NN and a $4 \times 7 \times 1$ NN by setting $N_i = 2$. In both figures, we use 3-bit RRAM and set $\sigma = 0.05$ in Eq. (9) for evaluation. The comparison between the trained function and the ideal function shows that each stage with low-precision RRAM can accurately approximate the ideal stage function with the aid of the proposed training framework.

IV. Experimental Results

A. Experimental Methodology

1) Training configuration: We set $N_i = 1, 2, 3$ to get three distinct resolution configurations in each pipeline stage in our experiments. For each stage, we train different NN models and each NN model is trained via stochastic gradient descent with the Adam optimizer using TensorFlow [15]. The weight precision A_R during training is set to be 1$\sim$7-bit. The batch size is 4096, and the projection step is performed every 256 iterations. We train for a total of 2×10^4 iterations for each sub-ADC model and residue model, varying the learning rate from 10^{-3} to 10^{-4} across the iterations.

2) Technology model: We use the HfO_x-based RRAM device model to simulate the crossbar array [16]. We set the resistance stochastic variation $\sigma = 0.05$, since it is a moderate variation based on the evaluations from prior work [17]. The transistor model is based on a standard $130nm$ CMOS technology. The inverters, output comparators, and transistor switches in the RRAM crossbars are simulated with the 130nm model using Cadence Spectre. The VTC group A_{VTC} is obtained by running 100 times Monte Carlo simulations. The simulation results presented in the following section are all based on SPICE simulation.

3) Metric of training accuracy: The trained accuracy of the sub-ADC/proposed NNADC is represented by the effective number of bits (ENOB)–a metric to evaluate the effective resolution of an ADC. We report ENOB based on its standard definition ENOB=(SNDR-1.76)/6.02, where the signal to noise and distortion ratio (SNDR) is measured from the sub-ADC's/proposed NNADC's output spectrum. The training accuracy of the residue circuit is represented by the mean-square error (MSE) between predicted residue function and ideal residue function. We report the MSE based on 2048 uniform sampling points in the full range of input $[0, V_{\text{DD}}]$.

B. Sub-block Evaluations

1) Resolution and robustness: To find a robust design for each stage, we study the relationship between the trained accuracy and RRAM precision of each sub-block with different NN sizes at a fixed stochastic variation. For these experiments, we first incorporate both CMOS PVT variations and limited precision of RRAM device into training, and then instantiate several batches of 100-run Monte Carlo simulations with a resistance variation $\sigma = 0.05$ in Eq. (9), and finally compute the median accuracy of each model.

We plot the trends in Fig. 5. Generally, an $(N_i + 1)$-bit RRAM precision is enough to train an NN model to accurately approximate an N_i-bit sub-ADC, which confirms the conclusion in previous work [7]. Particularly, larger size NN models with more hidden neurons can even accurately approximate an N_i-bit sub-ADC with N_i-bit RRAM precision. Similar conclusions can also be made from the trained performance of residue circuits. As the Fig. 5(b) shows, an $(N_i + 2)$-bit RRAM precision is enough to train an NN model to accurately approximate a residue circuit. Moreover, a larger size NN with more hidden layer neurons

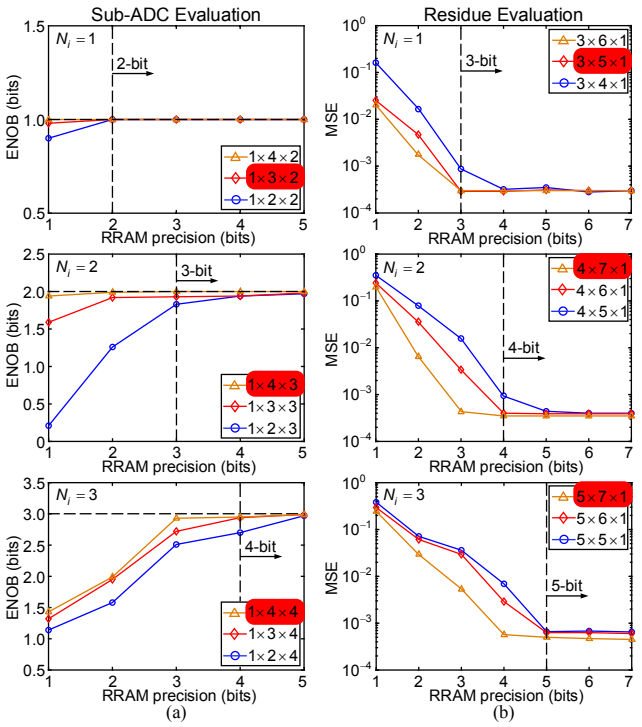

Fig. 5: Sub-block training performance using different NN models and RRAM precision at a fixed stochastic variation $\sigma = 0.05$. (a) The trend between ENOB and RRAM precision of sub-ADC under different NN models, where the N_i is set as 1, 2, 3 respectively. (b) The trend between MSE and RRAM precision of residue circuit under different NN models, where the N_i is set as 1, 2, 3 respectively.

can accurately approximate the residue circuit of N_i-bit stage with $(N_i + 1)$-bit RRAM precision.

2) Sub-block design trade-off: Each stage-i has design trade-off among power consumption P_i, sampling rate $f_{S,i}$ and area $A_{s,i}$. A completed design space exploration may involve the searching of different NN sizes of each sub-block in stage-i, RRAM precision and stochastic variations. Here, we use three pairs of sub-blocks highlighted by the solid boxes in Fig. 5 as an example to illustrate the design trade-off, since each of them shows enough accuracy and robustness with no more than 4-bit RRAM precision. For these experiments, we combine each pair of sub-blocks to form three distinct sub-blocks with resolution $N_i = 1, 2, 3$, respectively. We then fix the precision of RRAM device with 3-bit for for all building blocks except for the residue in $N_i = 3$ stage, which use 4-bit RRAM device. We finally study the relationship between the power E_j, speed f_j, and area A_j of each distinct stage-j ($j = 1, 2, 3$) by simulating the minimum power consumption/area of each distinct stage that works well at different sampling rates.

The trends are plotted in Fig. 6, which shows clear trade-offs between speed and power consumption, as well as speed and area, for each distinct stage. This is because in order to make each sub-block work well under faster speed, we need to increase the driving strength of the neurons by sizing up the inverters, which results in an increase of power consumption and area for each stage.

3) Design optimization: Based on the exploration of different sub-block configurations, an optimal design for the

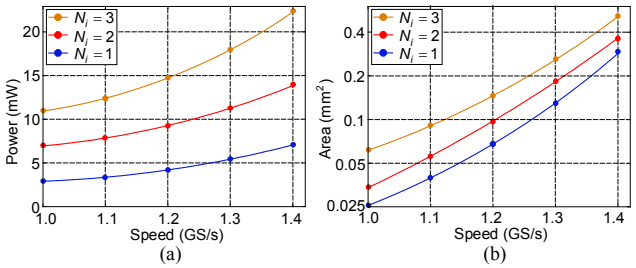

Fig. 6: Design trade-offs of three distinct stages, with resolution $N_i = 1, 2, 3$ respectively. (a) Power VS speed. (c) Area VS speed.

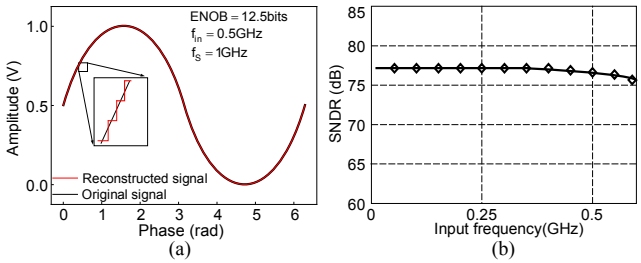

Fig. 7: (a) Reconstruction of a 14-bit pipelined NNADC with 3-bit RRAM whose pipelined chain consists of eleven stages: nine 1-bit stages, one 2-bit stage and one 3-bit sub-ADC. (b) SNDR trend.

proposed ADC with a given resolution can be derived by solving the following optimization problem:

$$\begin{aligned} \min \quad & FoM_W = P/(2^{ENOB} \cdot f_S) \\ \min \quad & A_{ADC} \\ s.t. \quad & \begin{cases} ENOB \le \sum_{i=1}^{M} N_i & N_i \in \{1,2,3\}, \\ P = \sum_{i=1}^{M} P_i & P_i \in \{E_1, E_2, E_3\}, \\ f_S = \min_{1 \le i \le M} \{f_{S,i}\} & f_{S,i} \in \{f_1, f_2, f_3\}, \\ A_{ADC} = \sum_{i=1}^{M} A_{s,i} & A_{s,i} \in \{A_1, A_2, A_3\}. \end{cases} \end{aligned} \tag{10}$$

Here, the first optimal objective FoM_W ($fJ/conv$) is a standard figure-of-merit that describes the energy consumption of one conversion for an ADC, and the second optimal objective A_{ADC} is the area of the proposed ADC. We set FoM_W as the main optimal objective, since energy efficiency usually is the most important consideration for most applications. In this way, as shown in Fig. 7, we can obtain an optimal design for a maximum 14-bit pipelined NNADC with 12.5 bits of ENOB, and 11.6$fJ/conv$ of FoM_W working at 1GS/s. It showcases the advantages of our proposed co-design framework that incorporates many circuit-level non-idealities in the training process, allowing us to realize a robust design cascading up to eleven stages, a level often unattainable with traditional pipelined ADCs.

C. Full Pipelined NNADC Evaluation

We choose the three distinct stages in Section IV-B to evaluate the quantization ability of the proposed full pipelined NNADC. We find that although the co-design framework can help us to train a low-resolution stage to approximate the ideal quantization function and residue function with high-fidelity, the minor discrepancy between the trained stage and ideal stage will propagate and aggregate along the pipeline and finally results in a wrong quantization. Our simulations based on various combinations of different pipeline stages show that a maximum 14-bit pipelined NNADC working at 1GS/s can be achieved by cascading nine 1-bit stages, one 2-bit stage and one 3-bit sub-ADC with 3-bit RRAM precision. Note that the last stage of the 14-bit pipelined NNADC does not need to generate residue. The reconstructed signal of this 14-bit ADC is shown in Fig. 7(a), where the ENOB is 12.5 bits under 1GHz sampling frequency. We also report the SNDR trend with input signal frequency in Fig. 7(b). The SNDR begins to degenerate after 0.5GHz input, verifying

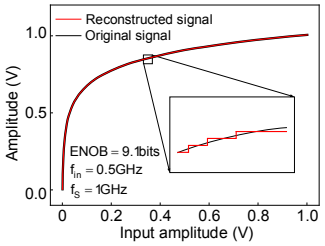

Fig. 8: A 10-bit logarithmic NNADC with ten 1-bit stages.

the sampling frequency ($\times 2$ of input signal frequency) of the proposed 14-bit NNADC is well above 1GHz.

Finally, we train a nonlinear ADC based on the same methodology using a logarithmic encoding on the input signal by replacing V_{in} in Eq. (3) with $V_{\text{in,log}} = V_{\text{DD}} \cdot \log_2(a+1)$ ($a \in \{0, 1\}$) to train a 1-bit stage. We find that a 10-bit logarithmic ADC with 9.1-bit ENOB working at 1GS/s can be achieved by cascading ten such 1-bit stages, and the reconstructed signal is illustrated in Fig. 8.

D. Performance Comparisons

1) Comparison with existing NNADCs: We first design an optimal 8-bit NNADC by cascading eight 1-bit stages in Section IV-B and compare it with previous NNADCs [6], [7]. The comparative data are summarized in the left columns of Table I. Compared with them, the proposed 8-bit NNADC can achieve the same resolution and higher energy efficiency with ultra-low precision 3-bit RRAM devices. Both NNADC1 and NNADC2 adopt a typical NN (Hopfield or MLP) architecture to directly train an 8-bit ADC without the optimization of architecture; therefore, they needs high-precision RRAM to achieve the targeted resolution of ADC. NNADC1 uses a large size ($1 \times 48 \times 16$) three-layer MLP as the circuits model, where parasitic aggregations on the large size crossbar array degenerates the conversion speed. In addition, more hidden neurons are used in NNADC1 which consume more energy. Since each stage in the proposed 8-bit NNADC resolves only 1-bit and has very small size, it can achieve faster conversion speed with higher energy-efficiency, and high-resolution with low-precision RRAM devices. Please note that the FoM_W reported in NNADC2 is based on sampling a low frequency (44KHz) signal at high frequency (1.66GHz). Therefore, it is considered outside the scope of a Nyquist ADC, and cannot be compared directly with our work on the same FoM_W basis.

2) Comparison with traditional nonlinear ADCs: We then compare the trained 10-bit logarithmic ADC with state-of-

TABLE I: Performance comparison with different types of ADCs.

ADC types	NNADC			Nonlinear ADC			Uniform ADC	
Work	NNADC1 [7]*	NNADC2 [6]*	**This work***	JSSC 09' [11]**	ISSCC 18' [3]**	**This work***	JSSC 15' [12]**	**This work***
Technology (nm)	130	180	**130**	180	90	**130**	65	**130**
Supply (V)	1.2	1.2	**1.5**	1.62	1.2	**1.5**	1.2	**1.5**
Area (mm^2)	0.2	0.005/0.01	**0.02**	0.56	1.54	**0.03**	0.594	**0.1**
Power (mW)	30	0.1/0.65	**25**	2.54	0.0063	**31.3**	49.7	**67.5**
f_S (S/s)	0.3G	1.66G/0.74G	**1G**	22M	33K	**1G**	0.25G	**1G**
Resolution (bits)	8	4/8	**8**	8	10	**10**	12	**14**
ENOB (bits)	7.96	3.7/(N/A)	**8**	5.68	9.5	**9.1**	10.6	**12.5**
FoM_W (fJ/c)	401	8.25/7.5	**97.7**	2380	263	**57**	108.5	**11.6**
RRAM precision	9	6/12	**3**	N/A	N/A	**3**	N/A	**3**
Reconfigurable ?	Yes	Yes/Yes	**Yes**	No	Yes	**Yes**	No	**Yes**

* The results are shown based on simulation.
** The results are shown on chip.

the-art traditional nonlinear ADCs [3], [11]. The comparative data are summarized in the middle columns of Table I. As it shows, the proposed 10-bit logarithmic ADC has competitive advantages in area, sampling rate, and energy efficiency. JSSC 09' [11] uses a pipelined architecture to implement an 8-bit logarithmic ADC. Due to the devices mismatch, its ENOB degenerates a bit from the targeted resolution. ISSCC 18' [3] requires >10-bit capacitive DAC to achieve a configurable 10-bit nonlinear quantization resolution; therefore, it can achieve high ENOB but only works at $\sim KS/s$ with significant area overhead. Since we adopt the proposed training framework to directly train a log-encoding signal using small-sized NN models and incorporating device non-idealities, we can achieve a logarithmic ADC with small area, high sampling rate and high ENOB.

3) Comparison with traditional uniform ADC: Finally, we compare the trained 14-bit uniform ADC with state-of-the-art traditional uniform ADC. The comparative data are summarized in the right columns of Table I. It shows that the proposed 14-bit NNADC has competitive advantages in sampling rate, ENOB, and energy efficiency. JSSC 15' [12] uses power hungry op-amps and dedicated calibration techniques, resulting in the power consumption overhead and degeneration of conversion speed. The proposed 14-bit NNADC uses low-resolution stages with very small NN size, enabling faster conversion speed with higher energy efficiency. The slight ENOB degeneration of the proposed ADC is caused by the discrepancy (between the trained stage and ideal stage) propagation along the pipeline stages. Also note that the performance of the proposed NNADCs and the performance of previous NNADCs are based on simulations, while the performance of the traditional nonlinear ADCs and uniform ADC are based on measurements.

V. Conclusion

In this paper, we present a co-design methodology that combines a pipelined hardware architecture with a custom NN training framework to achieve high-resolution NN-inspired ADC with low-precision RRAM devices. A systematic design exploration is performed to search the design space of the sub-ADCs and residue blocks to achieve a balanced trade-off between speed, area, and power consumption of each distinct low-resolution stages. Using SPICE simulation, we evaluate our design based on various ADC metrics and perform a comprehensive comparison of our work with different types of state-of-the-art ADCs. The comparison results demonstrate the compelling advantages of the proposed NN-inspired ADC with pipelined architecture in high energy efficiency, high ENOB and fast conversion speed. This work opens a new avenue to enable future intelligent analog-to-information interfaces for near-sensor analytics using NN-inspired design methodology.

Acknowledgement

This work was partially supported by the National Science Foundation (CNS-1657562).

References

[1] R. LiKamWa et al., "RedEye: Analog ConvNet Image Sensor Architecture for Continuous Mobile Vision," *IEEE ISCA*, 2016, pp. 255-266.
[2] B. Li et al., "RRAM-Based Analog Approximate Computing," *IEEE TCAD*, vol. 34, no. 12, pp. 1905-1917, 2015.
[3] J. Pena-Ramos et al., "A Fully Configurable Non-Linear Mixed-Signal Interface for Multi-Sensor Analytics," *IEEE JSSC*, vol. 53, no. 11, pp. 3140-3149, Nov. 2018.
[4] M. Buckler et al., "Reconfiguring the Imaging Pipeline for Computer Vision," *IEEE ICCV*, 2017, pp. 975-984.
[5] L. Gao et al., "Digital-to-analog and analog-to-digital conversion with metal oxide memristors for ultra-low power computing," *IEEE/ACM NanoArch*, 2013, pp. 19-22.
[6] L. Danial et al., "Breaking Through the Speed-Power-Accuracy Tradeoff in ADCs Using a Memristive Neuromorphic Architecture," *IEEE TETCI*, vol. 2, no. 5, pp. 396-409, Oct. 2018.
[7] W. Cao et al., "NeuADC: Neural Network-Inspired RRAM-Based Synthesizable Analog-to-Digital Conversion with Reconfigurable Quantization Support," *DATE*, 2019, pp. 1456-1461.
[8] T. F. Wu et al., "14.3 A 43pJ/Cycle Non-Volatile Microcontroller with 4.7μs Shutdown/Wake-up Integrating 2.3-bit/Cell Resistive RAM and Resilience Techniques," *IEEE ISSCC*, 2019, pp. 226-228.
[9] Y. Cai et al., "Training low bitwidth convolutional neural network on RRAM," *ASP-DAC*, 2018, pp. 117-122.
[10] H. -. P. Wong et al., "MetalOxide RRAM," *in Proceedings of the IEEE*, vol. 100, no. 6, pp. 1951-1970, June 2012.
[11] J. Lee et al., "A 2.5mW 80 dB DR 36dB SNDR 22 MS/s Logarithmic Pipeline ADC," *IEEE JSSC*, vol. 44, no. 10, pp. 2755-2765, Oct. 2009.
[12] H. H. Boo et al., "A 12b 250 MS/s Pipelined ADC With Virtual Ground Reference Buffers," *IEEE JSSC*, vol. 50, no. 12, pp. 2912-2921, 2015.
[13] Kurt Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, issue. 2, pp. 251-257, 1991.
[14] W. Cao et al., "NeuADC: Neural Network-Inspired Synthesizable Analog-to-Digital Conversion," *IEEE TCAD*, 2019, Early Access.
[15] Kingma et al, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.
[16] P. Chen and S. Yu, "Compact Modeling of RRAM Devices and Its Applications in 1T1R and 1S1R Array Design," *IEEE TED*, vol. 62, no. 12, pp. 4022-4028, Dec. 2015.
[17] B. Li, et al., "MErging the Interface: Power, area and accuracy co-optimization for RRAM crossbar-based mixed-signal computing system," *IEEE ACM/EDAA/IEEE DAC*, 2015, pp. 1-6.
[18] Weidong Cao et al., "A 40Gb/s 39mW 3-tap adaptive closed-loop decision feedback equalizer in 65nm CMOS," *IEEE MWSCAS*, 2015, pp. 1-4.