This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

Evaluating Neural Network-Inspired Analog-to-Digital Conversion with Low-Precision RRAM

Weidong Cao, *Student member, IEEE,* Liu Ke, *Student member, IEEE,*
Ayan Chakrabarti, *Member, IEEE,* and Xuan Zhang, *Member, IEEE*

***Abstract*—Recent work has demonstrated great potentials of neural network-inspired analog-to-digital converters (NNADCs) in many emerging applications. These NNADCs often rely on resistive random-access memory (RRAM) devices to realize basic NN operations, and usually need high-precision RRAM (6∼12-bit) to achieve moderate quantization resolutions (4∼8-bit). Such an optimistic assumption of RRAM precision, however, is not well supported by practical RRAM arrays in large-scale production process. In this paper, we evaluate two new designs of NNADC with low-precision RRAM devices. They take advantage of traditional two-stage/pipelined hardware architecture and a custom deep learning-based building block design methodology. Results obtained from SPICE simulations demonstrate a robust design of an 8-bit sub-ranging NNADC using 4-bit RRAM devices, as well as a 14-bit pipelined NNADC using 3-bit RRAM devices. The evaluations on the two NNADCs suggest that pipelined architecture is better to achieve higher-resolution using lower-precision RRAM. We also perform design space exploration on the building blocks of NNADCs to achieve a balanced performance trade-off. Comprehensive comparisons reveal improved power, speed performance, and competitive figure-of-merits (FoMs) of the pipelined NNADC, compared with state-of-the-art NNADCs and traditional ADCs. In addition, the proposed pipelined NNADC can support reconfigurable high-resolution nonlinear quantization with high conversion speed and low conversion energy, enabling intelligent analog-to-information interfaces for near-sensor processing.**

***Index Terms*—High-resolution ADC; Low-precision RRAM; Neural network; Nonlinear quantization.**

I. Introduction

MANY emerging applications have posed new challenges to design conventional analog-to-digital (A/D) converters (ADCs) [1–6]. For example, multi-sensor systems require nonlinear A/D quantization to maximize the extraction of useful features from raw analog signals, instead of the linearly uniform quantization performed by conventional ADCs [3, 5], because the nonlinear quantization scheme can alleviate the computational burden and reduce the power consumption of digital backend processing, which is the dominant bottleneck in intelligent multi-sensor systems. In addition, processing-in-memory (PIM) using non-volatile memory (NVM) crossbar arrays desires non-uniform quantization and adaptive tuning of ADCs to satisfy the specific bitline computation mechanisms [2, 10]. However, such flexible quantization schemes are not readily supported by conventional ADCs with fixed conversion references and thresholds.

To overcome these inherent limitations of conventional ADCs, recent works have introduced neural network-inspired ADCs (NNADCs) as a novel approach to designing flexible and intelligent A/D interfaces [7–14]. The basic idea behind NNADCs is that artificial neural networks (ANNs) can be trained to approximate the desirable quantization function of ADCs and these ANNs can be implemented on hardware circuits in the analog domain. For instance, a learnable 8-bit NNADC is presented to approximate multiple quantization schemes where the NN weights are trained off-line and can be reconfigured by programming the same hardware substrate [10, 11]. Another example is a 4-bit neuromorphic ADC proposed for general-purpose data conversion where the NN weights are on-line trained by leveraging the input signal amplitude statistics and application sensitivity [9]. These NNADCs are often built on resistive random-access memory (RRAM) crossbar array to realize the basic NN operations, with the potential to exceed the power-speed-accuracy trade-off in conventional ADC designs [9].

However, a major challenge to design such NNADCs is the limited conductance/resistance precision of the RRAM devices. Although measurement data from realistic RRAM fabrication process suggest the actual RRAM precision tends to be much lower (2∼4-bit) [15, 16], these NNADC designs often optimistically assume the availability of RRAM technology that can precisely program each cell with 6∼12-bit precision which translates to 2^6∼2^{12} distinctive conductance/resistance levels. In addition, the stochastic variation of RRAM can affect the NNADC's resolution. For example, on average the resolution of NNADCs degenerates 3-bit with 0.025 lognormal variation [30, 31] in previous works [10, 11]. Therefore, there exists a gap between the reality and the assumption of the RRAM precision, yet lacks a design methodology to build high-resolution NNADCs with low-precision RRAM devices.

In this paper, we explore to bridge this gap by evaluating two new designs of NNADC. They are implemented by combing the advantage of traditional sub-ranging/pipelined hardware architecture and a custom deep learning-based design methodology. The key idea of a sub-ranging/pipelined hardware architecture is that multiple consecutive low-resolution quantization stages can be cascaded into a two-stage/chain structure to obtain higher resolution, as long as the residue part of the signal can be amplified to full range and fed to the next

Manuscript received Dec 24, 2019; revised Apr 5, and Jun 20, 2020; accepted Jul 19, 2020.

W. Cao, L. Ke and X. Zhang are with the Department of Electrical and Systems Engineering, Washington University in St. Louis, St. Louis, MO, 63130 USA, e-mail: {weidong.cao@wustl.edu, ke.l@wustl.edu, xuan.zhang@wustl.edu}.

A. Chakrabarti is with the Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO, 63130 USA, e-mail: {ayan@wustl.edu}.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

quantization stage. Since each stage now only needs to resolve low-resolution[1], we can instantiate them on the hardware substrate with low-precision RRAM devices by accurately training NNs to approximate the ideal quantization functions and residue functions. Key innovations and contributions in this paper are as follow:

- We propose a deep learning-based design methodology to implement a general analog/mixed signal (AMS) circuit, which enables robust and efficient design of basic building blocks (e.g., sub-ADC, mixed-ADC and residue) in the sub-ranging ADCs and pipelined ADCs.
- We combine the sub-ranging/pipelined hardware architecture and the deep learning-based design methodology to achieve two new designs of NNADC: sub-ranging NNADC and pipelined NNADC. SPICE simulation results demonstrate that our proposed method enables the robust design of an 8-bit sub-ranging NNADC and a 14-bit pipelined NNADC using 4-bit RRAM and 3-bit RRAM, respectively. The evaluations on the two new designs suggest that the pipelined architecture is superior to achieve higher-resolution ADCs with lower-precision RRAM devices.
- We systematically evaluate the impacts of NN size and RRAM precision on the trained accuracy of the NN-inspired sub-ADC, mixed-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.
- Thorough comparisons among the pipelined NNADC, state-of-the-art NNADCs and traditional ADCs demonstrate competitive figure-of-merits (FoMs) of the proposed pipelined NNADC. Our proposed pipelined NNADC can also support reconfigurable high-resolution nonlinear quantization with high conversion speed and low conversion energy.

The rest of this paper is organized as follows. Section II provides preliminary backgrounds and related works on this research topic. A deep learning-based building block design methodology is proposed in Section III. The detailed implementation of building blocks is presented in Section IV. The designs of sub-ranging NNADC and pipelined NNADC are elaborated in Section V. Finally, we introduce the simulation methodology in Section VI and show the evaluation results in Section VII before concluding the paper in Section VIII.

II. Background And Related Work

To provide the background of our work, we first give a quick overview of the RRAM technology and how its crossbar architecture enables the efficient implementation of an ANN. We then briefly introduce some related works that have employed NN-inspired principles to realize A/D conversion and summarize the main challenges in current NNADC designs. Finally, we review some conventional ADCs, such as sub-ranging ADC and pipelined ADC, that use low-resolution stages to achieve high-resolution A/D quantization.

[1]∼5-bit for each stage in the two-stage architecture and 1∼3-bit for each stage in the pipelined architecture

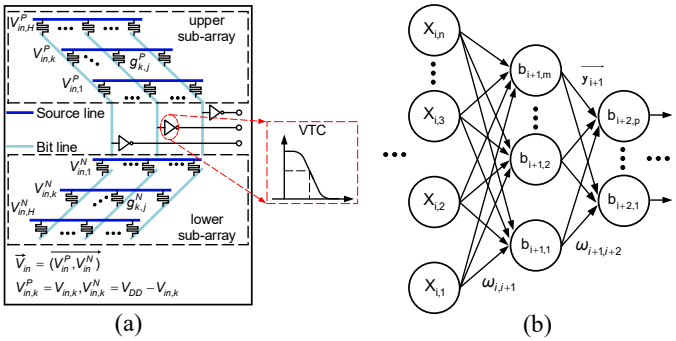


Fig. 1: (a) Hardware substrate to perform basic NN operations. The passive crossbar array composed of two sub-arrays executes VMM. The VTC of CMOS inverter acts as an NAF. (b) An example of a multi-layer ANN whose two adjacent layers are connected by weights.

A. RRAM Device, Crossbar Array and ANN

1) RRAM device: A RRAM device is a passive two-port element with variable resistance. It possesses many special advantages, such as small cell size ($4F^2$, F is the minimum feature size), excellent scalability ($<10nm$), and faster read/write time ($<10ns$) and better endurance ($\sim 10^{10}$ cycles) than Flash devices [2, 17–19].

2) RRAM crossbar array: RRAM devices can be organized into various ultra-dense crossbar array architectures [10, 21]. Fig. 1(a) shows a passive crossbar array, composed of two sub-arrays, to realize bipolar weights without using power-hungry operational-amplifiers (op-amps) [10, 11]. The relationship between the input voltage "vector" ($\vec{V}_{\text{in}}$) and the output voltage "vector" ($\vec{V}_{\text{o}}$) can be expressed as follows:

$$V_{\text{o},j} = \sum_{k=1}^{H} W_{k,j} \cdot V_{\text{in},k} + V_{\text{off},j}, \quad j \in \{1, 2, ..., M\}. \quad (1)$$

Here, k and j 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

$$W_{k,j} = \epsilon \cdot (g^U_{k,j} - g^L_{k,j}), \quad \epsilon = 1/\sum_{k=1}^{H} (g^U_{k,j} + g^L_{k,j}). \quad (2)$$

Therefore, the RRAM crossbar array can perform analog vector-matrix multiplication (VMM), and the parameters of the matrix rely on the RRAM resistance states. By configuring the passive crossbar arrays into a dual-path architecture as demonstrated in previous work [10, 11], a pair of complementary outputs can be obtained to feed as inputs to the next stage.

3) ANN: With the RRAM crossbar array, an ANN shown in Fig. 1(b) can be implemented on such hardware substrate. Generally, the ANN processes the data by executing the following operations layer-wise [34]:

$$\vec{y}_{i+1} = f(W_{i,i+1} \cdot \vec{x}_i + \vec{b}_{i+1}). \quad (3)$$

Here, $\vec{x}_i$ and $\vec{y}_{i+1}$ represent the data in the i^{th} and $(i+1)^{th}$ layer of the network. $W_{i,i+1}$ 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 inverter shown in Fig. 1(a) as follow

$$V_{\text{o},j} = \sigma_{\text{VTC}}(\sum_{k=1}^{H} W_{k,j} \cdot V_{\text{in},k} + V_{\text{off},j}). \quad (4)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

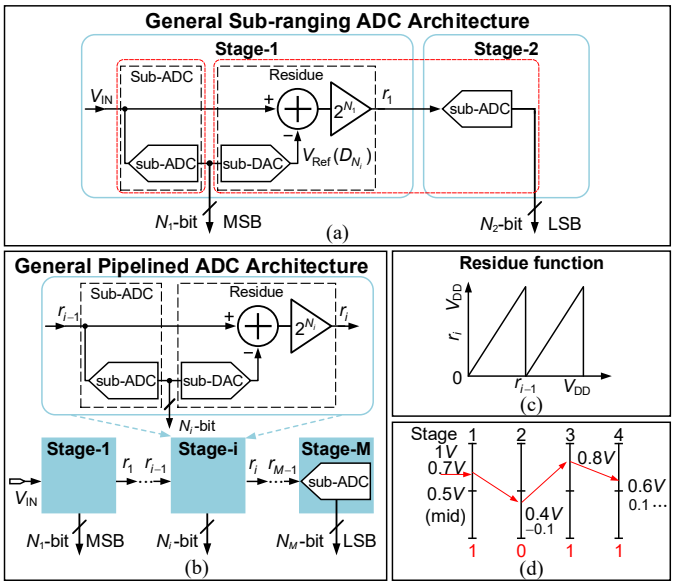


Fig. 2: Two well-established ADC topologies. (a) General architecture of sub-ranging ADC. (b) General architecture of pipelined ADC. (c) An example of the residue function when $N_i = 1$. (d) An example of a 4-bit pipelined ADC composed of four 1-bit stages.

Here, $\sigma_{\text{VTC}}(\cdot)$ is the voltage transfer characteristic (VTC) of the inverters. It can be used as an NAF [10, 20].

B. NNADCs

Analog-to-digital conversion can be viewed as a special case of classification problems, which maps a continuous analog signal to a series of multi-bit digital codes. An ANN can be trained to learn this input/output relationship, and its hardware implementation can be instantiated in the AMS domain. This is the basic idea behind NNADCs, that is to implement the learned ANN on a hardware substrate to approximate the desired quantization functions for data conversion:

$$\sum_{i=0}^{M-1} 2^i \cdot D_i = \text{round}\left(\frac{V_{\text{in}} - V_{\text{min}}}{V_{\text{max}} - V_{\text{min}}} \times (2^M - 1)\right). \quad (5)$$

Here, M is the resolution of ADC; V_{in} is an analog input and D_i is the i^{th} digital output bit of the digital code; 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 [7–13]. Although the NN architectures of these NNADCs vary from Hopfield NN to multi-layer perceptron (MLP), they all rely on a training process to learn the appropriate NN weights to accurately approximate flexible quantization schemes.

However, existing NNADCs often exhibit modest conversion resolution (4∼8-bit). Even worse, they invariably rely on optimistic assumption of RRAM precision (6∼12-bit) [7–13], which is not well substantiated by measurement data from realistic RRAM fabrication process [15, 16]. This resolution limitation severely constrains NNADCs' applications in many emerging multi-sensor systems that require >10-bit A/D interfaces or precise nonlinear conversion of analog signals for feature extraction and analog-to-information processing [1, 3, 5, 36]. In fact, it has been demonstrated in previous work [10] that training an A-bit ($A \leq 8$) quantization resolution with moderate conversion speed requires at least $(A+1)$-bit RRAM device. This conclusion suggests a direct trade-off between achievable ADC resolutions, NN sizes, and RRAM precisions [10].

C. Sub-ranging ADCs and Pipelined ADCs

The sub-ranging ADC shown in Fig. 2(a), and the pipelined ADC shown in Fig. 2(b) are well-established ADC topologies to achieve high sampling rate and high resolution with low-resolution quantization stages [22]. Usually, the sub-ranging ADCs have a two-stage architecture. Each stage resolves ∼5-bit quantization. The pipelined ADCs preserves a long chain structure with a significant pipeline delay. Each stage in the pipelined chain resolves 1∼3-bit quantization. Although they have different numbers of stages, each of these stages shares the same building blocks, e.g., sub-ADC and residue circuit.

The sub-ADC resolves an N_i-bit binary code 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-DAC by 2^{N_i} to generate the output residue r_i for next stage. This process can be expressed as a simple function:

$$r_i = [r_{i-1} - V_{\text{Ref}}(D_{N_i})] \cdot 2^{N_i}. \quad (6)$$

Here, $V_{\text{Ref}}(D_{N_i})$ 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 $V_{\text{Ref}}(0) = 0$ and $V_{\text{Ref}}(1) = V_{\text{DD}}/2$. And the corresponding residue function is shown in Fig. 2(c). Since each stage successively converts the analog input into its digital representation, the final outputs of the sub-ranging ADC and pipelined ADC are $(N_1 + N_2)$-bit and $\sum_{i=1}^{M} N_i$-bit digital codes, respectively. Note that N_i is not necessarily identical in all stages.

To understand the basic working principle of pipelined ADCs [22], we use a 4-bit pipelined ADC composed of four 1-bit stages as an example and illustrate the quantization steps in Fig. 2(d). Assuming the initial analog input is $0.7V$ ($V_{\text{DD}} = 1V$), then the sub-ADC in the first stage will output "1"—a digital code, and the residue block will output "$0.4V$"— an analog residue according to Eq. (6). The analog residue 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 $0.7V$ ($0.7/1 = 11.2/2^4 \approx 11/2^4$). To understand how residue is amplified in the stage with more than 1-bit resolution, we would suggest the readers to look at the Fig. 5 in Section IV.C.

III. Design Methodology of Building Blocks

To extend the architectures of sub-ranging ADC/pipelined ADC into NNADC's design, we first characterize their distinct building blocks in this section. We then demonstrate that these distinct building blocks can be universally described using a mathematical model of a general analog/mixed signal (AMS) circuit. Finally, we propose a deep learning-based framework to design the general AMS circuit, which enables robust and efficient implementation of basic building blocks in the sub-ranging ADCs and pipelined ADCs.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

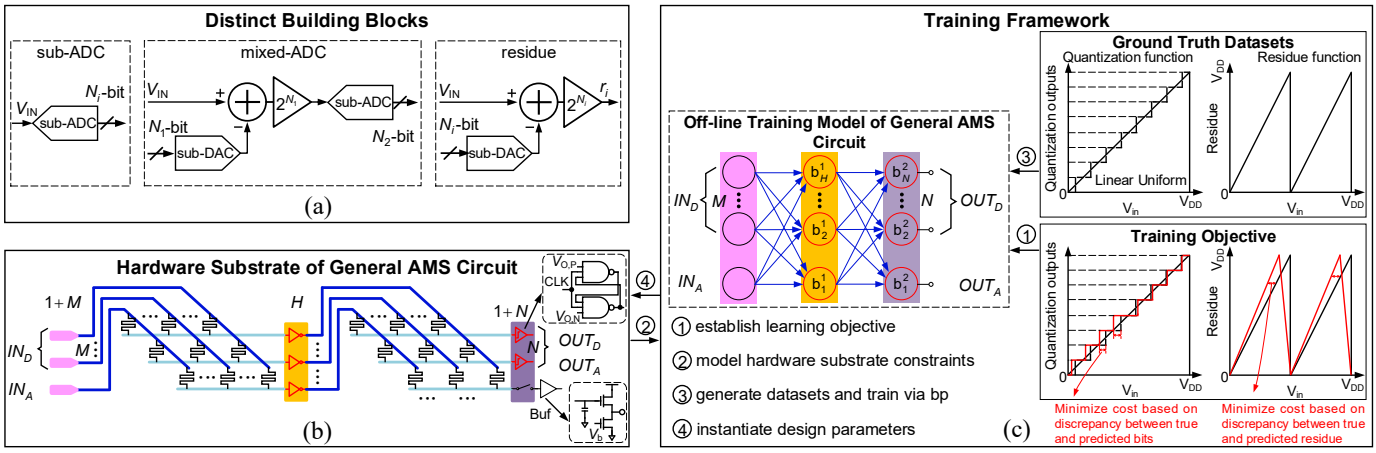


Fig. 3: Proposed deep learning-based design methodology. (a) Distinct building blocks in sub-ranging ADC and pipelined ADC. (b) Hardware substrate for the general AMS circuit. For simplicity, we do not show the extra input of each layer (extra row connected to $V_{\rm DD}$ or GND) for bias instantiation [11]. (c) Proposed training framework takes ground truth datasets as inputs during off-line training to find the optimal set of weights associated with the RRAM resistances to minimize the cost function and best approximate the ideal quantization function and residue function.

A. Characterization of Building Blocks

It can be observed in Fig. 2(b) that each stage (except for the last stage) in the pipelined ADC consists of two building blocks: sub-ADC and residue. The sub-ranging ADC has only two stages. A better way to characterize its distinct blocks is shown in the red dashed box in Fig. 2(a), which is to combine the residue in the first stage and the sub-ADC in the second stage. We name this block mixed-ADC, as it directly generates digital codes by using mixed signal inputs (initial analog input and the digital output from the sub-ADC in the first stage). This characterization has two advantages: 1) only two hardware NNs[2] are required to construct a sub-ranging NNADC instead of using three hardware NNs, saving hardware resources; 2) resolution can be improved compared with the sub-ranging NNADC constructed using three hardware NNs. In summary, there are totally three distinct building blocks in our design: sub-ADC, mixed-ADC, and residue, as illustrated in Fig. 3(a).

B. Mathematical Formulation of General AMS Circuits

The basic building blocks in Fig. 3(a) belong to a class of AMS circuit with specific input/output relationship. For example, the sub-ADC is an AMS circuit with analog input and digital output, whose ideal input/output relationship satisfies Eq. (5). Similarly, the residue block is an AMS circuit with mixed signal input and digital output, whose ideal input/output relationship satisfies Eq. (6). All these building blocks can be represented by a general AMS circuit whose inputs and outputs can be expressed as a simple mathematical function:

$$V_{\rm OUT} = f(V_{\rm IN}). \tag{7}$$

Here, $V_{\rm IN} = \{IN_{\rm A}, IN_{\rm D}\}$ are the mixed signal inputs of the circuit; $V_{\rm OUT} = \{OUT_{\rm A}, OUT_{\rm D}\}$ are the mixed signal outputs of the circuit. Note that the subscript "A" indicates "Analog", and "D" indicates "Digital". For instance, sub-ADC can be considered as a specific case of this general AMS circuit without $IN_{\rm D}$ and $OUT_{\rm A}$.

[2]As discussed in Section IV-A, each building block is built on a three-layer hardware substrate.

C. Deep Learning-Based Design Methodology

To evaluate the performance of NNADCs designed with two-stage and pipelined architecture, the first step is to form an effective design methodology for this type of general AMS circuit. Then each building block can be efficiently implemented as a specific case of the general AMS circuit. The design methodology contains two steps: hardware substrate and training framework, which are discussed as follows.

1) Hardware substrate: To implement the general AMS circuit, we use the RRAM crossbar array[3] and CMOS inverter illustrated in Fig. 1(a) as the hardware substrate. The corresponding hardware architecture is illustrated in Fig. 3(b). It preserves a three-layer NN architecture, because universal approximation theorem proves that a feed-forward three-layer NN with a single hidden layer can approximate arbitrary functions [25, 26]. As the Fig. 3(b) shows, the general AMS circuit has $(1 + M)$ input neurons (one analog input and M-bit digital inputs), and $(1 + N)$ output neurons (one analog output and N-bit digital outputs). Note that the hardware substrate can be generalized to both discrete-time systems and continuous-time systems. For the discrete-time systems, 3-input NAND gates [45] placed in the output layer are used to perform digitization while the sampling/hold (S/H) circuit is used as the "place holder" neuron for analog output to drive the next stage. A CMOS source follower-based S/H buffer circuit used in our design is shown in the inset of Fig. 3(b).

2) Training framework: We propose a hardware-oriented training framework for the general AMS circuit. It can accurately capture the circuit-level behavior of the hardware substrate and learn the associated hardware design parameters (e.g. RRAM conductance), to approximate the ideal input/output relationship of the general AMS circuit. The training framework possesses one important feature: non-idealities of devices, such as process, voltage and temperature (PVT) variations of CMOS device, and the limited precision of RRAM devices, can be incorporated into training to make the general AMS circuit robust to these defects [27]. This

[3]Each weight cell in the RRAM array consists of one transistor and one memristor (1T1R) and can operate in both compute mode and program mode. For simplicity, we use 1R cell to represent the practical 1T1R cell in this paper.

is the advantage of the NN-inspired design of general AMS circuits over the traditional design of AMS circuits, where, even with delicate calibration techniques, the non-idealities cannot be effectively mitigated [24]. The detailed training flow is shown in Fig. 3, which consists of the following four steps.

① *Learning objective construction:* The general AMS hardware substrate in Fig. 3(b) can be modeled as a three-layer NN:

$$\tilde{h} = L_1(V_{\text{IN}};\theta_1),\ h = \sigma_{\text{VTC}}(\tilde{h}),\ V_{\text{OUT}} = L_2(h;\theta_2). \quad (8)$$

Here, $V_{\text{IN}} = \{IN_{\text{A}}, IN_{\text{D}}\}$ are the mixed signal inputs for the AMS circuit. $\tilde{h}$ denote voltages at the output of the first crossbar layer. They are modeled as a linear function L_1 of V_{IN} with learnable parameters $\theta_1 = \{W_1, V_1\}$, corresponding to the weights and bias associated with the first layer required to be learned from the training. Each of these voltages is passed through an inverter, whose input-output relationship is modeled by a nonlinear function $\sigma_{\text{VTC}}(\cdot)$, to yield the vector h. The linear function L_2 models the second layer of the crossbar. It produces the output $V_{\text{OUT}} = \{OUT_{\text{A}}, OUT_{\text{D}}\}$ with learnable parameters $\theta_2 = \{W_2, V_2\}$, corresponding to the weights and bias associated with the second layer required to be learned from the training. The learning objective is to find optimal values for the parameters $\{\theta_1, \theta_2\}$ (corresponding to RRAM crossbar array conductances) such that for all values of V_{IN} in the input range, the circuit yields corresponding output V_{OUT} that are equal or close to the desired "ground truth" $V_{\text{OUT,GT}}$ in Eq. (7). Towards this goal, we define a cost function to measure the discrepancy between predicted V_{OUT} and true $V_{\text{OUT,GT}}$ based on the mean-square loss:

$$C(V_{\text{OUT}}, V_{\text{OUT,GT}}) = \sum\nolimits_j (f_{\text{disc}}(V_{\text{OUT,GT}}(j) - V_{\text{OUT}}(j)))^2. \quad (9)$$

Here, f_{disc} means various mathematical functions to measure discrepancy, such as L_2 norm, and cross-entropy. It is chosen depends on the practical learning objective.

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 as A_{VTC}, using the technology specification in Section VI. Meanwhile, we control the precision of weight with A_{R}-bit during the training. Finally, we let the summation of all elements (their absolute value) in each column ("0") of W_1 and W_2 be less than 1:

$$\sum(\text{abs}(W_1), 0) < 1; \quad \sum(\text{abs}(W_2), 0) < 1, \quad (10)$$

to reflect the weight constraints in Eq. (2).

③ *Hardware-oriented training:* We initialize the parameters $\{\theta_1, \theta_2\}$ randomly, and update them iteratively based on the gradients computed on the mini-batches of $\{(V_{\text{IN}}, V_{\text{OUT,GT}})\}$ pairs, which are randomly sampled from the input range. To incorporate the hardware constraints in step ② into training, we let each neuron j in Eq. (8) randomly pick up a VTC from A_{VTC} during training:

$$\sigma^j_{\text{VTC}} = A_{\text{VTC}}[f_{\text{randint}}(N_{\text{VTC}})], j = 1, 2, ..., H. \quad (11)$$

Here, $f_{\text{randint}}(N_{\text{VTC}})$ is a function to generate a random integer smaller than N_{VTC}. A detailed discussion of incorporating PVT variations into training can be found in our previous work [11]. We then periodically clip all values of W_1 between $[-1/(1+M), 1/(1+M)]$ to satisfy Eq. (10). To make W_2 satisfy the constraint in Eq. (10) as well, corresponding technique will be applied based on different training objectives. The details will be discussed in Section IV-B.

④ *Instantiate design parameters:* We adopt the same instantiation method in previous work [10], which is proven to always find a set of equivalent RRAM conductances for the trained weights. After this, we perturb each resistance *R* in the hardware substrate by:

$$R \leftarrow R \cdot e^{\theta};\quad \theta \sim \mathcal{N}(0, \sigma), \quad (12)$$

to evaluate the robustness of the NN model towards the stochastic variation of RRAM resistance [30, 31].

IV. Implementation of Building Blocks

After presenting the NN-inspired design methodology for the general AMS circuit, in this section, we elaborate how to implement the different building blocks. We first show the detailed hardware architecture of each distinct building block and then present the key training specifications based on their specific input/output relationship.

A. Hardware Implementation of Building Blocks

All distinct building blocks preserve a similar three-layer NN architecture and are implemented with the RRAM crossbar array and CMOS inverter illustrated in Fig. 1(a). Minor difference exists between different building blocks in NN size and the types of input/output neurons.

1) Sub-ADC: For 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 input layer, and $H_{F,i} \times S_i$ between the hidden and output layer, assuming there are $H_{F,i}$ and S_i neurons in the hidden and output layer, respectively. 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 our previous work [10], as it improves the training accuracy and reduces hidden layer size of the sub-ADC. To help the readers understand the concept of smooth encoding, we briefly re-clarify its definition here. The readers can refer our previous work [11] for details. Smooth codes represent each of the 2^{N_i} levels binary codes with S_i-bit unique codewords, adhering to two important principles. First, only one bit changes its value between two consecutive levels, a property similar to "Gray codes". Second, each bit in S_i-bit unique codewords (2^{N_i} levels) flips a minimum number of times. Given a group of parameters N_i and S_i, the S_i-bit codewords start with an all-zero bits codeword for the lowest level in the 2^{N_i} unique levels, and then flip the bit that was least recently flipped for each subsequent level. For example, we use $3 \rightarrow 2$ smooth encoding to train a 2-bit sub-ADC with 3-bit smooth codes as output in Fig. 5(a). A one-to-one mapping between a 3-bit smooth code and a 2-bit binary code is "000 → 00", "001 → 01", "011 → 10", and "111 → 11".

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

2) Mixed-ADC: For mixed-ADC, there are $(1 + S_1)$ input neurons (one analog input and S_1 digital inputs), and S_2 output neurons. Therefore, the weight matrix dimensions are $H_2 \times (1 + S_1)$ between the hidden and input layer and $H_2 \times S_2$ between the hidden and output layer, assuming H_2 hidden neurons. Note that the digital output S_2 is also a smooth code.

3) Residue: For residue, there are $(1 + S_i)$ input neurons (one for analog input and S_i for digital inputs), and only one analog output neuron. Therefore, the weight matrix dimensions are $H_{R,i} \times (1 + S_i)$ between the hidden and input layer, and $H_{R,i} \times 1$ between the hidden and output layer, assuming there are $H_{R,i}$ hidden neurons. Note that since the op-amps and comparators in Fig. 2 are eliminated in the NN-inspired design of sub-ADC, mixed-ADC and residue, considerable power saving can be obtained from each stage.

B. Training of Building Blocks

We focus on describing some key specifications for training mixed-ADC and residue, as similar strategies for training sub-ADCs have been elaborated in previous work [10]. The main procedures to train a mixed-ADC and a residue follow the steps in Section III-C, but have some modifications in step ① and step ③ based on different learning objectives.

1) Mixed-ADC: For mixed-ADC, its output is an S_2-bit smooth digital code; therefore, its hardware substrate can be modeled by adapting Eq. (8) as follows:

$$\begin{aligned} &\tilde{h} = L_1(V_{\text{in}}, D_{S_1}; \theta_1), \quad h = \sigma_{\text{VTC}}(\tilde{h}), \\ &\tilde{D}_{S_2} = L_2(h; \theta_2), \quad D_{S_2} = \tilde{D}_{S_2} > 0. \end{aligned} \tag{13}$$

Here, D_{S_1} indicates the digital output from the sub-ADC in previous stage ("1" means V_{DD}, and "0" means GND). The final output bit-vector D_{S_2} is obtained by thresholding: yielding 0 for each element of $\tilde{D}_{S_2}$ that is below 0, and yielding 1 otherwise. The learning objective is to find optimal values of parameters $\{\theta_1, \theta_2\}$ such that for all values of $\{(V_{\text{in}}, D_{S_1})\}$ in the input range, the circuit yields corresponding digital output D_{S_2} that are equal or close to the desired "ground truth" D_{GT} in Eq. (5). To achieve this aim, the cost function in Eq. (9) can be adapted using the following cross-entropy loss:

$$\begin{aligned} C(\tilde{D}_{S_2}, D_{\text{GT}}) = \sum_{i=1}^{M} [&D_{\text{GT}i} \log(1 + e^{-\tilde{D}_{S_2,i}}) + \\ &(1 - D_{\text{GT}i}) \log(1 + e^{\tilde{D}_{S_2,i}})]^2. \end{aligned} \tag{14}$$

To make the second layer weight W_2 satisfy the constraint in Eq. (10), we first normalize both W_2 (and proportionally V_2) such that the sum of all the elements (their absolute value) across the same column is less than magnitude 1:

$$W_2' = W_2/\alpha, \quad V_2' = V_2/\alpha. \tag{15}$$

Here, $\alpha = \beta \cdot \sum(\text{abs}(W_2), 0)$ is a normalization coefficient. $\sum(\text{abs}(W_2), 0)$ represents the summation of all elements (their absolute value) in the same column. $\beta > 1$ is a scaling factor.

2) Residue: For residue block, its output is an analog value; therefore, the hardware substrate can be modeled as

$$\begin{aligned} &\tilde{h}_i = L_1(r_{i-1}, D_{S_i}; \theta_{1,i}), \quad h_i = \sigma_{\text{VTC}}(\tilde{h}_i), \\ &r_i = L_2(h_i; \theta_{2,i}). \end{aligned} \tag{16}$$

Here, i is the index of stage-i ($i \in \{1, \ldots, M\}$); D_{S_i} indicates the digital output of the sub-ADC; r_{i-1} is the scalar residue input of stage-i. The learning objective is to find optimal values for $\{\theta_{1,i}, \theta_{2,i}\}$ such that for all 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. (6). To achieve this aim, the cost function in Eq. (9) can be adapted as

$$C(r_i, r_{\text{GT}}) = \sum\nolimits_j [r_{\text{GT}}(j) - r_i(j)]^2. \tag{17}$$

We find that when $N_i = 1, 2$, the residue function can be trained to the full range by periodically clipping all values of W_2 between $[-1/H_{R,i}, 1/H_{R,i}]$ to satisfy Eq. (10). However, the same method is invalid when applied to train the residue function of $N_i = 3$. As the last row of Fig. 5(b) shows, the residue function of $N_i = 3$ is highly nonlinear which is hard to be accurately approximated by training a moderate size NN with constrained W_2. Therefore, during the training, we do not put any constraints on W_2 such that a moderate size NN can be trained to accurately approximate the residue function. After training, we use the same method shown in Eq. (15) to normalize the trained W_2 to make it satisfy Eq. (10). Although this method also results in the scaled predicted residue range[4], the following sub-ADC can be accurately trained to quantize this analog signal with scaled range. The last row of Fig. 5 gives an example that even with an input dynamic range as low as ∼0.1V, the NN can still yield 3-bit quantization.

C. Examples of Trained Building Blocks

During the training, we tried to train various pairs of sub-ADC and mixed-ADC for sub-ranging NNADC, and different pairs of sub-ADC and residue for pipelined NNADC. We find that, the mixed-ADC can be trained to accurately approximate a maximum 3-bit resolution with a moderate size NN. In addition, residue is hard to be accurately trained using a moderate size NN when $N_i \geq 4$. Here, we show three pairs of sub-ADC and mixed-ADC with different resolutions ($N_1 = 3, 4, 5$, and $N_2 = 2, 2, 3$), and three pairs of sub-ADC and residue block with different resolutions ($N_i = 1, 2, 3$) using the simulation methodology described in Section VI.

Since our designs are based on a dual-path architecture to perform "pseudo differential" operation, we evaluate the trained performance of building blocks by using the input ranges when the positive input voltage is higher than the negative one. Fig. 4 illustrates the SPICE simulation of different trained pairs in sub-ranging NNADC. The sub-ADCs in Fig. 4(a) are trained through a $1 \times 4 \times 4$, $1 \times 10 \times 8$, and $1 \times 10 \times 10$ NN, respectively; while the mixed-ADCs in Fig. 4(b) are trained through a $5 \times 6 \times 3$, $9 \times 10 \times 4$, and $11 \times 12 \times 6$ NN, respectively. In both figures, we use 4-bit RRAM device and set $\sigma = 0.05$ in Eq. (12) for evaluation. Note that we only show a small fraction ($[0, 1/2^{N_1}]), N_1 = 3, 4, 5$) of the reconstructed signal in the full input range[5].

[4]The dynamic range of the predicted residue function changes from $[0, V_{\text{DD}}]$ to $[V_{\text{DD}}/2 - V_{\text{DD}}/(2\alpha), V_{\text{DD}}/2 + V_{\text{DD}}/(2\alpha)]$.

[5]For each fraction $[j/2^{N_1}, (j+1)/2^{N_1}]$, where $N_1 = 3, 4, 5; j = 0, 1, 2, ..., 2^{N_1} - 1$, in the full range of the input signal, the reconstructed signal shows almost the same shape. We just show the reconstructed signal in $[0, 1/2^{N_1}]$ as an example.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

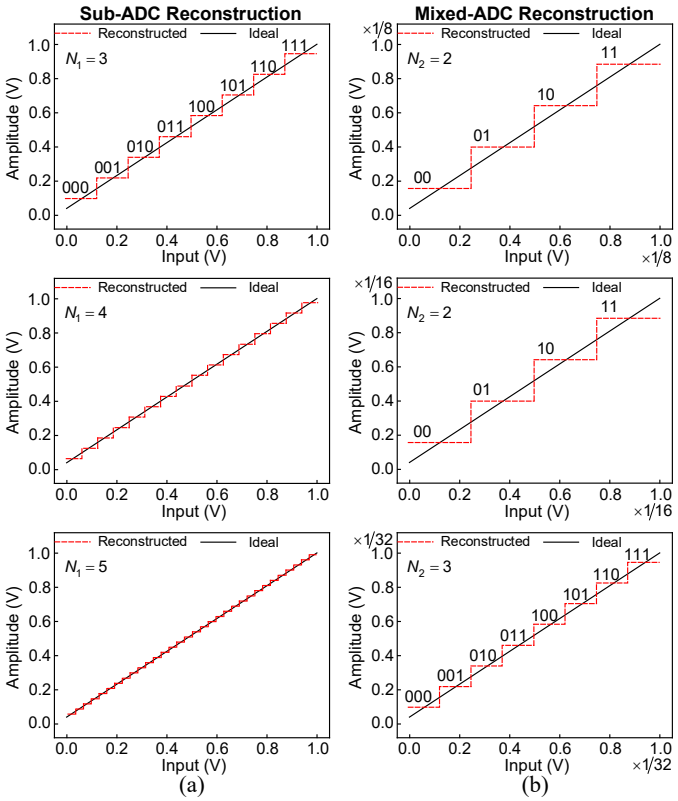


Fig. 4: Illustrations of trained sub-ADC and mixed-ADC with different resolutions in sub-ranging NNADC. (a) Sub-ADC ($N_1 = 3, 4, 5$). (b) Mixed-ADC ($N_2 = 2, 2, 3$).

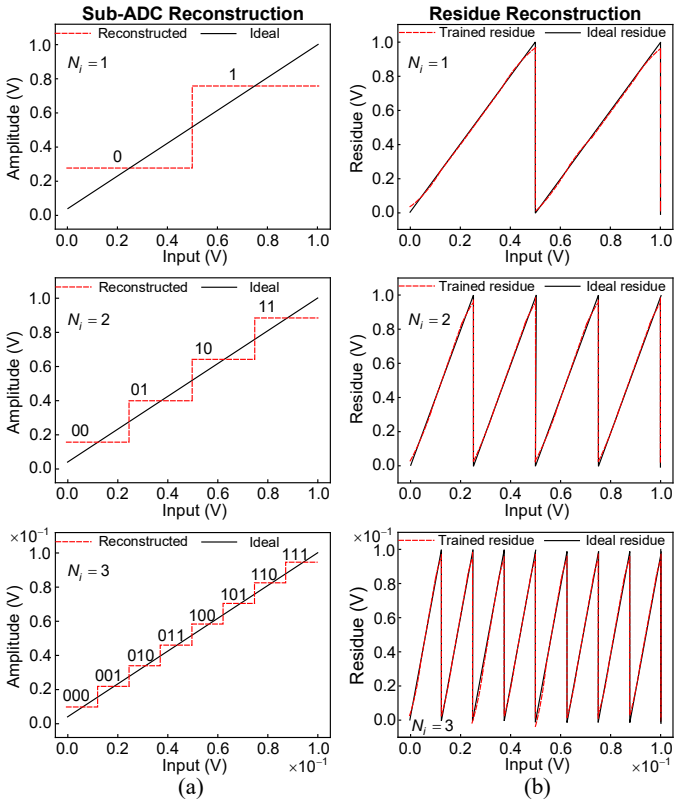


Fig. 5: Illustrations of trained sub-ADC and residue functions for a pipeline stage with different resolutions. (a) Sub-ADC ($N_i = 1, 2, 3$). (b) Residue ($N_i = 1, 2, 3$).

Fig. 5 illustrates the SPICE simulation of different trained pairs of pipelined NNADC. The sub-ADCs in Fig. 5(a) are trained through a $1 \times 3 \times 2$, $1 \times 4 \times 3$, and $1 \times 4 \times 4$ NN, respectively; while the residue blocks in Fig. 5(b) are trained through a $3 \times 5 \times 1$, $4 \times 7 \times 1$, and $5 \times 7 \times 1$ NN, respectively. In both figures, we use 3-bit RRAM device and set $\sigma = 0.05$ in Eq. (12) for evaluation. The comparison between the trained function and the ideal function shows that each pair with low-precision RRAM can accurately approximate the ideal stage function with the aid of the proposed design methodology. Note that: 1) the signal reconstructions of sub-ADC and mixed-ADC are based the method proposed in our previous work [11]; 2) the reconstructed signal of sub-ADC is not applied as $V_{\text{Ref}}(D_{N_i})$ in Eq. (6).

V. Implementation of NN-Inspired ADCs

In this section, we employ the NN-inspired building blocks implemented in previous section into the traditional two-stage/pipelined architecture to construct the sub-ranging NNADC and the pipelined NNADC. We first introduce the system level hardware architecture of these NNADCs. Then, we show some system level training strategies to improve the performance of these NNADCs. Finally, we present the advantages of co-design that combines NN-inspired design methodology with traditional two-stage/pipelined architecture.

A. Hardware Architecture of Full NNADCs

The hardware architecture of the proposed sub-ranging NNADC is presented in Fig. 6(a), where two three-layer NNs are adopted in the full NNADC design, and each of them can be mapped into the corresponding sub-ADC and mixed-ADC shown in Fig. 2(a). Similarly, the overall architecture of the proposed pipelined NNADC is presented in Fig. 6(b), where a pipelined architecture of cascaded conversion stages is adopted in the design. For stage-i in the proposed pipelined NNADC, we use two three-layer NNs to implement it, and each of them can be mapped into the corresponding sub-ADC and residue block shown in Fig. 2(b). A digital combiner designed by simple sequential circuits is used to synchronize each blocks' output to achieve the total resolution for the proposed sub-ranging NNADC and pipelined NNADC.

B. Training Strategy of Full NNADCs

To improve the performance of full NNADCs, an important technique used in our design is the collaborative (end-to-end) training of building blocks. For the sub-ranging NNADC, as illustrated in Fig. 3(a), the inputs of mixed-ADC include the original analog signal V_{IN} and the N_1-bit digital outputs from the previous sub-ADC. Therefore, the sub-ADC is first trained to approximate the ideal quantization function with high-fidelity, then its digital outputs and original analog inputs are used as ground truth data to train mixed-ADC. Similarly, for each stage in the pipelined NNADC, the sub-ADC is initially trained to approximate the ideal quantization function with high-fidelity, then its digital outputs and original analog inputs are directly used as ground truth data to train residue block. Compared with the independent design of building blocks in traditional ADCs, the collaborative training flow can

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

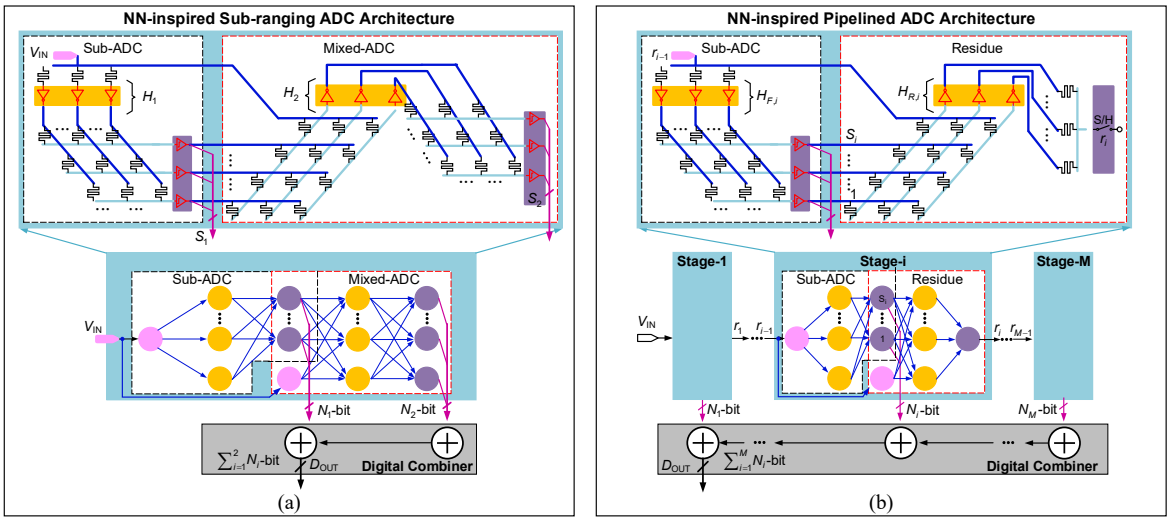


Fig. 6: Hardware architectures of the proposed NNADCs. (a) Sub-ranging NNADC. (b) Pipelined NNADC. For simplicity, we do not show the extra input of each layer (extra row connected to V_{DD} or GND) for bias instantiation [11].

effectively minimize the discrepancy between the training or circuit artifacts and the ideal conversion at each stage.

C. Co-design Analysis

The co-design of combining two-stage/pipelined architecture and deep learning-based design methodology brings two direct benefits. First, each stage in the proposed NNADCs now only needs to resolve low-resolution quantization (∼5-bit for each stage in sub-ranging NNADC, and 1∼3-bit for each stage in pipelined NNADC), which can be well achieved within the precision limit of current RRAM fabrication process [15, 16]. With the aid of the training framework in Section III, we can also automatically derive the optimal design of the low-resolution stages [10]. Second, although many cascading stages are needed in the pipelined NNADC, there only exist three distinct low-resolution configurations to choose for each stage, namely $N_i = 1, 2, 3$. This allows us to simplify the design process by focusing on optimizing the sub-blocks of each stage. The full pipelined system can then be assembled by iterating through different combinations of the building blocks with different resolution configurations.

VI. Simulation Methodology

In this section, we present the detailed methodology used in our simulation setup to train, design, and evaluate the proposed NNADCs. We first summarize the configurations used in our training setup, and then present the technology model to design the hardware substrate. Finally, we introduce the metrics to evaluate the trained accuracy of each building blocks.

A. Training Configuration

We set $N_1 = 3, 4, 5$ and $N_2 = 2, 2, 3$ to get three pairs of sub-ADC and mixed-ADC for sub-ranging NNADC, and set $N_i = 1, 2, 3$ to get three pairs of sub-ADC and residue for each stage in pipelined NNADC. For each pairs, we train different NN models. Each NN model is trained via stochastic gradient descent with Adam optimizer [28] using TensorFlow [29]. The moderate size ($N_{\text{IN}} \times N_H \times N_O$) of each NN model is constrained by $N_{\text{IN}} \leq 12$, $N_H \leq 12$, and $N_O \leq 10$ based on previous work [2]. We incorporate both CMOS PVT variations and the limited precision A_{R} of RRAM device into training. The weight precision A_{R} during training is set to be 1∼7-bit [42]. The batch size is 4096, and the projection step is performed every 256 iterations on $W_i, i = 1, 2$. We train a total of $2{\times}10^4$ iterations for each sub-ADC, mixed-ADC, and residue model, varying the learning rate from 10^{-3} to 10^{-4} across the iterations. The training time for each block is generally less than 10 minutes on a single TITAN GPU.

B. Technology Model

We use the HfO_x-based RRAM device model to simulate the crossbar array [32, 33]. Since we use the passive crossbar array [10] to achieve VMM, and the input analog signal has small amplitude, the voltage drop across the device is small; therefore, the $I{-}V$ relationship of the RRAM can be considered as linear in our work[6]. We use non-overlapping linearly spaced RRAM conductance to build each weight cell. We choose a moderate variation $\sigma = 0.05$ in our evaluation from a broad range of RRAM literature [27, 34, 35, 40, 41], which is equivalent to $\pm 15\%$ in 3σ range. RRAM endurance can be up to 10^{10} according to previous works [2, 17–19]. Although the retention time of different RRAM devices can vary from hundreds of ms to years [46–53] especially under extreme operating temperatures, most state-of-the-art works [48–52] show that they can ensure 10-year retention without conductance drifting at $85°C$. Therefore, the NNADC is able to handle typical applications over a long period and can also be calibrated by reprogramming the device as long as the endurance is still in its working range in spite of any long-term drifts. 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

[6]The nonlinearity of RRAM due to large crossing voltage will result in the computation error of RRAM crossbar array, degenerating the resolution of NNADC. One can replicate more devices in each cell and connect them serially to reduce the voltage drop of each RRAM.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

TABLE I: Simulation configuration parameters.

Training Parameters	**Description**
Optimizer	Adam
Batch size	4096
Projection step	256
Number of iterations	2×10^4
Learning rate	$[10^{-3},10^{-4}]$
A_{R} (RRAM precision)	1∼7-bit
N_1 (bits)	$3, 4, 5$
N_2 (bits)	$2, 2, 3$
N_i (bits)	$1, 2, 3$
Technology Parameters	**Description**
CMOS technology (nm)	CMOS $130nm$
Process variation	ss/tt/ff/sf/fs
Voltage variation	$1.47V \sim 1.53V$
Temperature variation	$-40°C \sim 80°$C
RRAM technology	HfOx-based RRAM
RRAM tunneling gap (nm)	0.2∼1.9
RRAM resistance range	290Ω ∼ 500kΩ
RRAM resistance variation (σ)	0.05

A_{VTC} is obtained by running Monte Carlo simulations 100 times using the methodology in our previous work [11]. Due to the small size of RRAM crossbar array and the short distance of the connection between layers, the wire parasitic resistance is negligible and can be reasonably ignored. We perform extensive SPICE simulations to determine the optimal inverter sizing to ensure sufficient driving strength under the worst-case loading effect from the subsequent layer. Configuration parameters from both the training setup and the technology model are summarized in Table I.

C. Metric of Trained Accuracy

The accuracy of sub-ADC, mixed-ADC and NNADC is represented by the effective number of bit (ENOB)—a metric to evaluate the effective resolution of an ADC. We report ENOB based on its standard definition ENOB $= (\mathrm{SNDR}-1.76)/6.02$, where the signal to noise and distortion ratio (SNDR) is calculated from the following equation:

$$\mathrm{SNDR} = 10 \cdot \log_{10}\Big(\frac{\sum_{i=1}^{N}\big(V_{\mathrm{Rec}}(t_i)\big)^2}{\sum_{i=1}^{N}\big(V_{\mathrm{IN}}(t_i)-V_{\mathrm{Rec}}(t_i)\big)^2}\Big). \quad (18)$$

Here, V_{IN} is the original input signal; V_{Rec} is the reconstructed signal based on the digital bits from SPICE simulation; and the samples are performed across multiple clock periods. The trained accuracy of the residue block is represented by the mean-square error (MSE) between the predicted residue function and ideal residue function. We report the MSE based on 2048 uniform sampling points in the full range of input. The power, differential non-linearity (DNL), integral non-linearity (INL), and max conversion speed are obtained from the SPICE simulation. Specially, the INL and DNL are calculated based on the simulation data according to their standard definitions.

VII. Evaluation Results

In this section, we perform comprehensive evaluations on sub-ranging NNADC and pipelined NNADC. We start to compare the building blocks of two NNADCs. We then perform design space exploration to find optimal stage configuration in each NNADC with balanced trade-off between speed, area, and power consumption, based on which we also investigate the trade-off between these two NNADCs. Since pipelined NNADC has greater potential to achieve higher-resolution with lower-precision RRAM, we finally evaluate the performance of the proposed pipelined NNADC with various state-of-the-art ADCs, such as NNADCs, nonlinear ADCs, and conventional pipelined ADCs.

A. Block-level Comparisons

We first investigate the relationship between the trained accuracy and RRAM precision of each building block with different NN sizes. In these simulations, we 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. (12), and finally compute the median trained accuracy of each model.

We plot such trends for the building blocks of two NNADCs in Fig. 7 and Fig. 8, respectively. Generally, an $(N_1 + 1)$-bit ((N_2+1)-bit) RRAM precision is enough to accurately train an NN model to approximate an N_1-bit sub-ADC (N_2-bit mixed-ADC) in sub-ranging NNADC, which conforms with the conclusion in previous work [10]. Particularly, larger size NN models with more hidden layer neurons and output neurons can even accurately approximate an N_1-bit sub-ADC (N_2-bit mixed-ADC) with N_1-bit (N_2-bit) RRAM precision. Similar conclusions can also be made from the trained accuracy of building blocks in pipelined NNADC. As the Fig. 8 shows, an (N_i+1)-bit ((N_i+2)-bit) RRAM precision is enough to train an NN model to accurately approximate an N_i-bit sub-ADC (residue block). Moreover, a larger size NN with more hidden layer neurons can accurately approximate the residue circuit of N_i-bit stage with $(N_i + 1)$-bit RRAM.

However, the comparison between the sub-ADC of sub-ranging NNADC and the sub-ADC of pipelined NNADC shows that training ≥4-bit sub-ADC with low-precision RRAM requires a larger size NN. The reason is that the non-linearity of the quantization function (Eq. (5)) becomes more evident[7] as the resolution of sub-ADC increases. To approximate such highly nonlinear functions, a larger size NN with more neurons is required. It can also be observed that the mixed-ADC following the sub-ADC can resolve only 2∼3-bit quantization even with a large size NN. This is because mixed-ADC actually includes two functions (e.g., residue function and sub-ADC quantization function). It can achieve only low resolutions even if a large size NN is applied to approximate such complex functions. However, it is worth noting that when $N_1 \leq 3$, both the sub-ADC and the following mixed-ADC can be accurately approximated with small size NNs and low precision RRAM (3-bit), which indicates that a two-stage architecture of sub-ranging ADC is better to achieve ≤5-bit NNADCs with fewer stages and simpler hardware structure.

Previous works [7–14] show that the total units of an NNADC with the size of $1 \times (N_H \times N_O$ scale with the targeted resolution N in a cubic trend ($(N_H \times N_O) \sim N^3$). Here, N_H is the number of hidden units, which is usually

[7]Least significant bit (LSB) of sub-ADC will flip 2^M times according to Eq. (5) during the total quantization levels.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

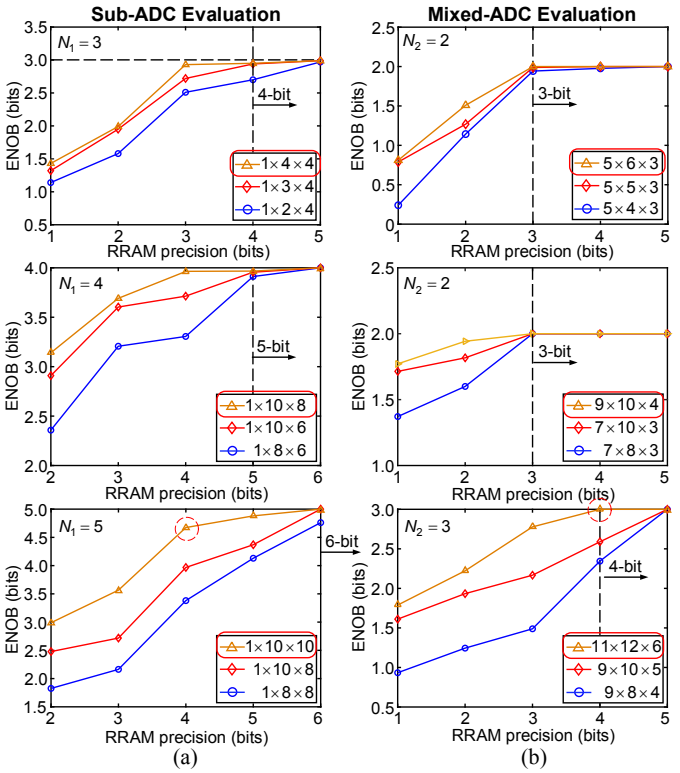


Fig. 7: Building block training performance using different NN models and RRAM precision at a fixed stochastic variation $\sigma = 0.05$ in sub-ranging NNADC. Note that each row is a pair. (a) The trend between ENOB and RRAM precision of sub-ADC under different NN models, where the $N_1 = 3, 4, 5$. (b) The trend between ENOB and RRAM precision of mixed-ADC under different NN models, where the $N_2 = 2, 2, 3$.

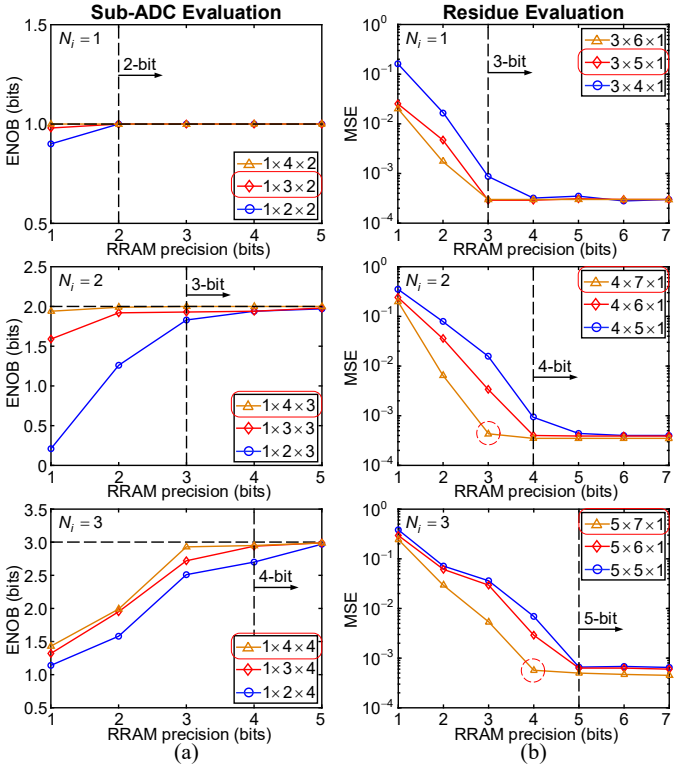


Fig. 8: Building block training performance using different NN models and RRAM precision at a fixed stochastic variation $\sigma = 0.05$ in pipelined NNADC. Note that each row is a pair. (a) The trend between ENOB and RRAM precision of sub-ADC under different NN models, where the N_i is set as 1, 2, 3. (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.

proportional to N^2; and N_O is the number of output neurons. The similar trends can also be observed from the building blocks shown in Fig. 7 and Fig. 8, where the size of sub-ADC and residue cubically scales with the resolution. Such a relationship provides a first order estimation for the required total units (T_{Units}) to achieve an N-bit NNADC:

$$\begin{cases} T_{\text{Units}} \sim \sum_{i=1}^{M} (N_i)^3, \\ N = \sum_{i=1}^{M} N_i. \end{cases} \tag{19}$$

Here, M is the number of stages required for the pipelined NNADC and N_i is the resolution of each stage.

B. Design Exploration

1) Design Trade-off of Building Blocks: Based on the study of building blocks in Section VII-A, we can design high-fidelity low-resolution stages with small size NNs to achieve: 1) a moderate resolution sub-ranging NNADC in a two-stage architecture, and 2) a high-resolution pipelined NNADC by combining different low-resolution stages in a pipelined chain. However, 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 involves the searching of different NN sizes of each building block in stage-i, RRAM precision and stochastic variations. Here, we use one pair of building blocks in the first row of Fig. 7, and three pairs of building blocks in Fig. 8 as an example to illustrate the design trade-off. Note that each of them (highlighted in red solid boxes) shows enough accuracy and robustness with no more than 4-bit RRAM precision. For the sub-ranging NNADC, each building block is a distinct stage which has resolution $N_1 = 3$ and $N_2 = 2$ respectively. For the pipelined NNADC, we combine each pair of building blocks in Fig. 8 to form three distinct stages with resolution $N_i = 1, 2, 3$, respectively. During the simulation, we fix the precision of RRAM device at 3-bit for all building blocks except for the residue in $N_i = 3$ stage, where a 4-bit RRAM is used. We finally study the relationship between the power (E), speed (f), and area (A) of each distinct stage of two NNADCs by simulating the minimum power consumption/area of each distinct stage that works well at different sampling rates.

The trends are plotted in Fig. 9, which shows clear trade-offs between speed and power consumption, as well as speed and area, for each distinct stage. In order to make each building block work well under faster speed, we need to increase the driven strength of the neurons by sizing up the inverters, which results in an increase of power consumption and area of each stage. A further comparison shows that at the same resolution, the distinct stage of sub-ranging NNADC is more energy-efficient and has smaller area than the distinct stage of pipelined NNADC. The benefits come from the simpler implementation of each stage in the sub-ranging NNADC, where the residue is not required to be approximated.

2) NNADCs design trade-off: Sub-ranging NNADC has fewer stages and simpler implementation of each stage. Pipelined NNADC has more stages and more complex im-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

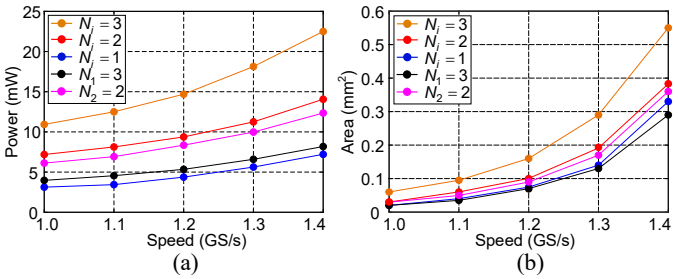


Fig. 9: Design trade-offs of three distinct stages ($N_i = 1, 2, 3$) in pipelined NNADC and two stages ($N_1 = 3$ and $N_2 = 2$) in sub-ranging NNADC. (a) Power VS speed. (c) Area VS speed.

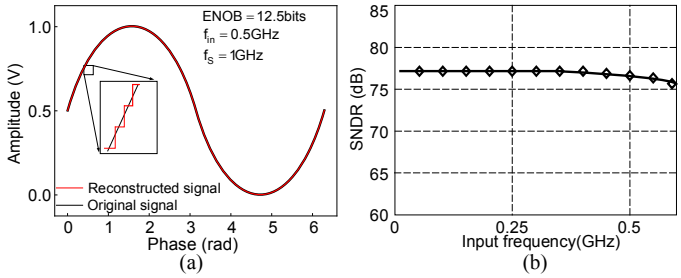


Fig. 10: (a) Reconstruction of a 14-bit pipelined NNADC with 3-bit RRAM whose pipelined chain consists of eleven stages: nine 1-bit stages, one 2-bit stage and one 3-bit sub-ADC. (b) SNDR trend of the proposed NNADC.

plementation of each stage. Therefore, there exists trade-off between these two NNADCs. To make a fair comparison, we first fix the precision of RRAM device at 3-bit. Under this condition, the sub-ranging NNADC can achieve a maximum 5-bit resolution by cascading the 3-bit sub-ADC and 2-bit mixed-ADC shown in Fig. 7. We find that to achieve the same 5-bit resolution, sub-ranging NNADC is more energy-efficient and has smaller area no matter how the pipelined NNADC combines its low-resolution stages. We then relax the the precision of RRAM device to 4-bit and explore the maximum resolution that sub-ranging NNADC can achieve. Our SPICE simulations show that an 8-bit sub-ranging NNADC with 7.3 bits ENOB can be achieved by combining the 5-bit sub-ADC and 3-bit mixed-ADC (highlighted in red solid box) shown in Fig. 7. Conversely, although each stage of pipelined NNADC resolves only 1∼3-bit quantization, it can achieve a much higher resolution by cascading many lower-resolution stages. As shown in Section VII-C, we can achieve a 14-bit pipelined NNADC by cascading nine 1-bit stages, one 2-bit stage and one 3-bit sub-ADC and using 3-bit RRAM.

In conclusion, with 3-bit RRAM, sub-ranging NNADC has higher energy-efficiency and smaller area to achieve a low-resolution (≤5-bit) NNADC, while pipelined NNADC is a better architecture to achieve high-resolution (≥6-bit) NNADC whose maximum resolution is 14-bit by cascading more low-resolution stages. In the following sections, we focus on evaluating the pipelined NNADC due to its higher-resolution.

3) Design optimization: Based on the exploration of different building block configurations, an optimal design for the proposed pipelined NNADC with a given resolution can be derived by solving the following optimization problem:

$$\min \quad (1)\ FoM_W = P/(2^{ENOB} \cdot f_S); \quad (2)\ A_{\text{ADC}}.$$
$$s.t. \begin{cases} ENOB \le \sum_{i=1}^{M} N_i & N_i \in \{1,2,3\}, \\ P = \sum_{i=1}^{M} P_i & P_i \in \{E_1, E_2, E_3\}, \\ f_S = \min\limits_{1\le i\le M} \{f_{S,i}\} & f_{S,i} \in \{f_1, f_2, f_3\}, \\ A_{\text{ADC}} = \sum_{i=1}^{M} A_{s,i} & A_{s,i} \in \{A_1, A_2, A_3\}. \end{cases} \tag{20}$$

Here, the first optimal objective FoM_W (fJ/c) is a standard figure-of-merit (FoM) 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. 10, we can obtain an optimal design for a maximum 14-bit pipelined NNADC with 12.5 bits of ENOB, 11.6fJ/c of FoM_W working at 1$GS/$s. It showcases the advantages of our proposed co-design methodology that by incorporating the consideration of many circuit-level non-idealities in the training process, it allows 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 chose the three distinct stages (highlighted in the red solid boxes) in Fig. 8 to evaluate the quantization ability of the proposed full pipelined NNADC. We find that although the NN-inspired design methodology 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; therefore, the pipelined stages cannot be infinite in a practical design.

Our simulations based on various combinations of different pipeline stages show that a maximum 14-bit pipelined NNADC working at 1$GS/$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. 10(a), where the ENOB is 12.5 bits under 1GHz sampling frequency. We then show the SNDR trend with input signal frequency in Fig. 10(b). The SNDR begins to degenerate after the input frequency goes beyond 0.5GHz, verifying the sampling frequency (×2 of input signal frequency) of the 14-bit NNADC is well above 1GHz. We also report the differential non-linearity (DNL) and integral non-linearity (INL) of the proposed NNADC. It is simulated at a typical-typical CMOS process corner after one-time instantiation on RRAM substrate with a fixed lognormal variation $\sigma = 0.05$. The DNL is $+0.71/-0.42$LSB (least significant bit) and the INL is $+0.98/-0.25$LSB, in the normal range of the traditional ADCs (e.g., DNL$\in [-1, 1]$LSB, INL$\in [-1, 1]$LSB). To show its robust performance, we perform 100 Monte Carlo simulations on the proposed 14-bit NNADC by setting $\sigma = 0.05$. The median ENOB we are able to obtain is ∼12.1-bit. We also perform extensive Monte Carlo simulations to capture the PVT effects from the CMOS devices and compensate its negative impact using variation-aware training [11]. The result indicates an ENOB centered around 12 bits can be achieved by the proposed NNADC with high robustness.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2020.3013563, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

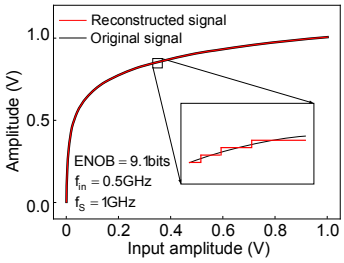


Fig. 11: A 10-bit logarithmic NNADC with ten 1-bit stages.

Finally, we train a nonlinear ADC based on the same methodology proposed in previous work [11] using a logarithmic encoding on the input signal by replacing V_{in} in Eq. (5) with $V_{\text{in,log}} = \log_2(a+1)$ ($a \in [0,1]$) to train a 1-bit stage. We find that a 10-bit logarithmic ADC with 9.1-bit ENOB working at $1GS/\text{s}$ sampling rate can be achieved by cascading ten such 1-bit stages. The reconstructed signal of this 10-bit ADC is illustrated in Fig. 11. Note that other quantization mechanisms can also be achieved based on previous work [11].

D. Performance Comparisons

1) Comparison with existing NNADCs: We first design an optimal 8-bit NNADC by cascading eight 1-bit stages (highlighted in the red solid boxes) in Fig. 8 and compare it with previous NNADCs [9, 10]. The comparative data are summarized in the left columns of Table II. NNADC1 [10], NNADC2 [9] are two representative NNADCs. Compared with them, the proposed 8-bit NNADC can achieve the same resolution with extremely low precision RRAM devices (3-bit) and high energy efficiency. Both NNADC1 and NNADC2 adopt a typical NN (MLP for NNADC1, and Hopfield for NNADC2) 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. NNADC2 uses $1 \times \frac{N \cdot (N+1)}{2} \times N$ size to achieve an N-bit ADC. Since each stage in the proposed 8-bit NNADC resolves only 1-bit and has very small size ($1 \times 3 \times 2$ for sub-ADC and $3 \times 5 \times 1$ for residue block), it can achieve faster conversion speed with higher energy-efficiency, and high-resolution with low-precision RRAM devices. Since each stage in the proposed 8-bit NNADC resolves only 1-bit and has a 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) input 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 and nonlinear NNADC: We then compare the 10-bit logarithmic ADC trained using our proposed method and presented in Section VII-C with state-of-the-art traditional nonlinear ADCs [3, 23]. The comparative data are summarized in the middle columns of Table II. Compared with state-of-the-art nonlinear ADCs, the proposed 10-bit logarithmic ADC has competitive advantages in area, sampling rate, and energy efficiency. JSSC09' [23] uses a pipelined architecture to implement an 8-bit logarithmic ADC. Due to the devices mismatch of switched-capacitors, the ENOB of [23] degenerates 2.3 bits from the targeted resolution. JSSC18' [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 ∼KHz with significant area overhead. Since we adopt the proposed training framework to directly train a log-encoding signal considering small-sized NN models and incorporating device non-idealities, we can achieve a logarithmic ADC with small area, high sampling rate and high ENOB. NNADC3 [43] is a recent work by dedicating the RRAM conductance to realize logarithmic quantization function. Compared with this work, our proposed NNADC can achieve higher resolution using lower-precision devices with improved performance.

3) Comparison with traditional uniform pipelined ADC and pipelined NNADC: We also compare the 14-bit uniform ADC in Section VII-C with state-of-the-art traditional uniform ADC. The comparative data are summarized in the right columns of Table II. Compared with JSSC15' [24], the proposed 14-bit NNADC has competitive advantages in sampling rate, ENOB, and energy efficiency. JSSC15' uses power hungry op-amps and dedicated calibration techniques, resulting in the overhead of power consumption and degeneration of conversion speed. NNADC4 [44] is a recent work which uses two-stage architecture to achieve a pipelined ADC. It can achieve 7.6 bits EONB with 6-bit RRAM. 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.

4) Special 1-bit example: Finally, since RRAM is still an emerging device with many active research and development efforts, we would like to provide a projection here by studying the performance of NNADCs with pure 1-bit RRAM in the design. We choose the two-stage architecture to design a 5-bit sub-ranging NNADC whose performance is listed in the fifth column of Table II. It shows that even with a pure 1-bit RRAM, we still can achieve an accurate NNADC with moderate performance.

In summary, by taking the advantages of traditional pipelined architecture and the NN-inspired design methodology, we can not only use low-resolution RRAM devices to achieve high-resolution NNADCs whose performance are superior to state-of-the-art ADCs, but also can realize versatile quantization schemes on the same hardware substrate which can be easily configured for different applications, such as near sensing data processing, in-memory computing bases on NVM crossbar array.

TABLE II: Performance comparison with different types of ADCs.

ADC types	NNADC[a]				Nonlinear ADC				Uniform ADC		
Work	NNADC1 [10]	NNADC2 [9]	**This work**[de]	Special 1-bit example[b]	JSSC09' [23][c]	JSSC18' [3][c]	NNADC3 [43][a]	**This work**[ad]	JSSC15' [24][c]	NNADC4 [44][a]	**This work**[ad]
Technology (nm)	130	180	**130**	130	180	90	180	**130**	65	180	**130**
Power supply (V)	1.2	1.8	**1.5**	1.5	1.62	1.2	1.8	**1.5**	1.2	1.8	**1.5**
Area (mm^2)	0.2	0.0049/0.01	**0.02**	0.18	0.56	1.54	N/A	**0.03**	0.594	N/A	**0.1**
Power (mW)	30	0.1/0.65	**25**	24	2.54	0.0063	0.045	**31.3**	49.7	0.272	**67.5**
f_S (S/s)	0.3G	1.66/0.74G	**1G**	0.4G	22M	33K	100K	**1G**	0.25G	1.66G	**1G**
Resolution (bits)	8	4/8	**8**	5	8	10	3	**10**	12	8	**14**
ENOB (bits)	7.96	3.7/(N/A)	**8**	4.91	5.68	9.5	2.55	**9.1**	10.6	7.6	**12.5**
FoM_W (fJ/c)	401	8.25/7.5	**97.7**	1.996×10^5	2380	263	77.2×10^3	**57**	108.5	0.97	**11.6**
RRAM precision	9	6/12	**3**	**1**	N/A	N/A	6	**3**	N/A	6	**3**
Configurable ?	Yes	Yes	**Yes**	Yes	No	Yes	Yes	**Yes**	No	Yes	**Yes**

[a] The results are shown based on simulation.
[b] The sub-ADC is trained via a $1 \times 12 \times 9$ NN which has 3-bit quantization resolution. The mixed-ADC is trained via a $10 \times 10 \times 8$ NN which has 2-bit quantization resolution.
[c] The results are based on measurement.
[d] The area of the proposed NNADCs does not include peripheral circuits whose area overhead can be mitigated through sharing among all devices [38]. The programming power is not counted in our evaluation, either.
[e] Note that the proposed NNADC is based on eight-stage pipelined architecture, which has 8 cycles of latency.

VIII. Conclusion

In this paper, we combine the sub-ranging/pipelined hardware architecture and the deep learning-based building block design methodology to achieve two new designs of NNADC. A systematic design exploration is also performed to search the design space of building blocks to achieve a balanced trade-off between speed, area, and power consumption of each distinct low-resolution stages for the NNADCs. The evaluations between the two new designs of NNADC suggest that pipelined architecture is superior to achieve higher-resolution with lower-precision RRAM. We also 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. This work opens a new avenue to enable future intelligent analog-to-information interfaces for near-sensor analytics and processing using NN-inspired design methodology.

Acknowledgment

This work was partially supported by USA National Science Foundation under grant No. CNS-1657562 and CCF-1942900.

References

[1] R. LiKamWa, Y. Hou, Y. Gao, M. Polansky and L. Zhong, "RedEye: Analog ConvNet Image Sensor Architecture for Continuous Mobile Vision," *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, Seoul, 2016, pp. 255-266.

[2] B. Li, P. Gu, Y. Shan, Y. Wang, Y. Chen and H. Yang, "RRAM-Based Analog Approximate Computing," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 34, no. 12, pp. 1905-1917, Dec. 2015.

[3] J. Pena-Ramos, K. Badami, S. Lauwereins and M. Verhelst, "A Fully Configurable Non-Linear Mixed-Signal Interface for Multi-Sensor Analytics," in *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 53, no. 11, pp. 3140-3149, Nov. 2018.

[4] M. Judy, A. M. Sodagar, R. Lotfi and M. Sawan, "Nonlinear Signal-Specific ADC for Efficient Neural Recording in Brain-Machine Interfaces," in *IEEE Transactions on Biomedical Circuits and Systems (TBioCAS)*, vol. 8, no. 3, pp. 371-381, June 2014.

[5] M. Buckler, S. Jayasuriya and A. Sampson, "Reconfiguring the Imaging Pipeline for Computer Vision," *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, 2017, pp. 975-984.

[6] S. Angizi, Z. He, A. Awad and D. Fan, "MRIMA: An MRAM-based In-Memory Accelerator," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*. Early express.

[7] Y. Xu, C. S. Thakur, T. J. Hamilton, J. Tapson, R. Wang and A. van Schaik, "A reconfigurable mixed-signal implementation of a neuromorphic ADC," *2015 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Atlanta, GA, 2015, pp. 1-4.

[8] L. Gao et al., "Digital-to-analog and analog-to-digital conversion with metal oxide memristors for ultra-low power computing," *2013 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, Brooklyn, NY, 2013, pp. 19-22.

[9] L. Danial, N. Wainstein, S. Kraus and S. Kvatinsky, "Breaking Through the Speed-Power-Accuracy Tradeoff in ADCs Using a Memristive Neuromorphic Architecture," *in IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 5, pp. 396-409, Oct. 2018.

[10] W. Cao, X. He, A. Chakrabarti, X. Zhang, "NeuADC: Neural Network-Inspired RRAM-Based Synthesizable Analog-to-Digital Conversion with Reconfigurable Quantization Support," *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Florence, Italy, 2019, pp. 1456-1461.

[11] W. Cao, X. He, A. Chakrabarti, X. Zhang, "NeuADC: Neural Network-Inspired Synthesizable Analog-to-Digital Conversion," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2019, Early Access.

[12] X. Guo et al., "Modeling and Experimental Demonstration of a Hopfield Network Analog-to-Digital Converter with Hybrid CMOS/Memristor Circuits," *Frontiers in Neuroscience*, vol. 9, no. 488, pp. 1-8, Dec. 2015.

[13] A. Fayyazi, M. Ansari, M. Kamal, A. Afzali-Kusha and M. Pedram, "An Ultra Low-Power Memristive Neuromorphic Circuit for Internet of Things Smart Sensors," in *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1011-1022, April 2018.

[14] W. Cao, L. Ke, A. Chakrabarti, X. Zhang, "Neural Network-Inspired Analog-to-Digital Conversion to Achieve Super-Resolution with Low-Precision RRAM Devices," *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, Westminster, CO, 2019, arXiv:1911.12815.

[15] 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," *2019 IEEE International Solid- State Circuits Conference (ISSCC)*, San Francisco, CA, USA, 2019, pp. 226-228.

[16] Y. Cai et al., "Training low bitwidth convolutional neural network on RRAM," *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jeju, 2018, pp. 117-122.

[17] H. -. P. Wong et al., "Metal–Oxide RRAM," *in Proceedings of the IEEE*, vol. 100, no. 6, pp. 1951-1970, June 2012.

[18] P. Chi et al., "PRIME: A Novel Processing-in-Memory Architecture for Neural Network Computation in ReRAM-Based Main Memory," *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, Seoul, 2016, pp. 27-39.

[19] Y. Zha, J. Li, "Liquid Silicon-Monona: A Reconfigurable Memory-Oriented Computing Fabric with Scalable Multi-Context Support,". *2018 ACM Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, New York, 2018, pp. 214-228.

[20] B. Karlik et al, "Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks," *IJAE*, vol. 1, no. 4, pp. 111-122, 2011.

[21] P. Chen, X. Peng and S. Yu, "NeuroSim: A Circuit-Level Macro Model for Benchmarking Neuro-Inspired Architectures in Online Learning," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 37, no. 12, pp. 3067-3080, Dec. 2018.

[22] R. Harjani, "http://people.ece.umn.edu/ harjani/courses/8331/ADC-pipeline_lecture.PDF".

[23] J. Lee et al., "A 2.5mW 80 dB DR 36dB SNDR 22 MS/s Logarithmic Pipeline ADC," *in IEEE Journal of Solid-State Circuits (JSSC)*, vol. 44, no. 10, pp. 2755-2765, Oct. 2009.

[24] H. H. Boo, D. S. Boning and H. Lee, "A 12b 250 MS/s Pipelined ADC With Virtual Ground Reference Buffers," *in IEEE Journal of Solid-State Circuits (JSSC)*, vol. 50, no. 12, pp. 2912-2921, Dec. 2015.

[25] Kurt Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, issue. 2, pp. 251-257, 1991.

[26] Y. Ito, "Approximation Capability of Layered Neural Networks with Sigmoid Units on Two Layers," in *Neural Computation*, vol. 6, no. 6, pp. 1233-1243, Nov. 1994.

[27] L. Chen et al., "Accelerator-friendly neural-network training: Learning variations and defects in RRAM crossbar," *2017 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Lausanne, 2017, pp. 19-24.

[28] Kingma et al, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[29] M. Abadi et al, "TensorFlow: A system for large-scale machine learning," *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016, pp. 265-283.

[30] S. R. Lee et al., "Multi-level switching of triple-layered TaOx RRAM with excellent reliability for storage class memory," *2012 Symposium on VLSI Technology (VLSIT)*, Honolulu, HI, 2012, pp. 71-72.

[31] F. Bedeschi, R. Fackenthal, C. Resta, et al, "A bipolar-selected phase change memory featuring multi-level cell storage," *IEEE Journal of Solid State Circuits (JSSC)*, vol. 44, no. 1, pp. 217-227, 2009.

[32] P. Chen and S. Yu, "Compact Modeling of RRAM Devices and Its Applications in 1T1R and 1S1R Array Design," *in IEEE Transactions on Electron Devices (TED)*, vol. 62, no. 12, pp. 4022-4028, Dec. 2015.

[33] S. Yu et al., "A Low Energy Oxide-Based Electronic Synaptic Device for Neuromorphic Visual Systems with Tolerance to Device Variation," *Advanced Materials*, vol. 25, pp. 1774-1779, Mar. 2013.

[34] B. Li, et al., "MErging the Interface: Power, area and accuracy co-optimization for RRAM crossbar-based mixed-signal computing system," *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, San Francisco, CA, 2015, pp. 1-6.

[35] Y. Long, X. She, S. Mukhopadhyay, "Design of Reliable DNN Accelerator with Un-reliable ReRAM," *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Florence, Italy, 2019, pp. 1-6.

[36] M. Verhelst and A. Bahai, "Where Analog Meets Digital: Analog-to-Information Conversion and Beyond," *in IEEE Solid-State Circuits Magazine*, vol. 7, no. 3, pp. 67-80, Summer 2015.

[37] M. Prezioso et al, "Training and operation of an integrated neuromorphic network based on metal-oxide memristors," *Nature*, vol. 521, no. 7550, pp. 61-64, 2015.

[38] A. Levisse, B. Giraud, J. P. Noël, M. Moreau and J. M. Portal, "Sneak Path compensation circuit for programming and read operations in RRAM-based Cross Point architectures," *2015 15th Non-Volatile Memory Technology Symposium (NVMTS)*, Beijing, 2015, pp. 1-4.

[39] Y. Deng et al., "RRAM Crossbar Array With Cell Selection Device: A Device and Circuit Interaction Study," in *IEEE Transactions on Electron Devices*, vol. 60, no. 2, pp. 719-726, Feb. 2013.

[40] A. Chen, "Utilizing the Variability of Resistive Random Access Memory to Implement Reconfigurable Physical Unclonable Functions," *IEEE Electron Device Letters*, vol. 36, no. 2, pp. 138-140, Feb. 2015.

[41] B. Liu et al, "Vortex: Variation-aware training for memristor X-bar," *IEEE/ACM Design Automation Conference (DAC)*, San Francisco, CA, 2015, pp. 1-6.

[42] F. Alibart et al, "High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," *Nanotechnology*, vol. 23, no. 7, 075201, 2012.

[43] L. Danial, K. Sharma, S. Dwivedi, and S. Kvatinsky, "Logarithmic Neural Network Data Converters using Memristors for Biomedical Applications," *Proceedings of the IEEE Biomedical Circuits and Systems (BioCAS)*, 2019.

[44] L. Danial, Kanishka Sharma, and Shahar Kvatinsky, "A Pipelined Memristive Neural Network Analog-to-Digital Converter," *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020.

[45] S. Weaver, B. Hershberg and U. Moon, "Digitally Synthesized Stochastic Flash ADC using only Standard Digital Cells," *IEEE Transactions on Circuits and Systems, Part I: Regular Papers, (TCAS-I)*, vol. 61, no. 1, pp. 84-91, 2014.

[46] K. Maeda, S. Matsuda, K. Takeuchi and R. Yasuhara, "Observation and Analysis of Bit-by-Bit Cell Current Variation During Data-Retention of TaOx-based ReRAM," 2018 48th European Solid-State Device Research Conference (ESSDERC), Dresden, 2018, pp. 46-49.

[47] R. Berdan, C. Lim, A. Khiat, C. Papavassiliou and T. Prodromakis, "A Memristor SPICE Model Accounting for Volatile Characteristics of Practical ReRAM," in IEEE Electron Device Letters, vol. 35, no. 1, pp. 135-137, Jan. 2014.

[48] C. H. Cheng, A. Chin and F. S. Yeh, "Novel Ultra-low power RRAM with good endurance and retention," 2010 Symposium on VLSI Technology, Honolulu, 2010, pp. 85-86.

[49] T. Chang, S. H. Jo, and W. Lu, "Short-term memory to long-term memory transition in a nanoscale memristor," ACS Nano, vol. 5, no. 9, pp. 7669–7676, 2011.

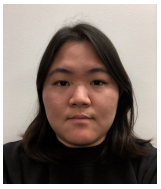
[50] Z. Wei et al., "Demonstration of high-density ReRAM ensuring 10-year retention at 85°C] based on a newly developed reliability model," 2011 International Electron Devices Meeting, Washington, DC, 2011, pp. 31.4.1-31.4.4.

[51] M. Azzaz et al., "Endurance/Retention Trade Off in HfOx and TaOx Based RRAM," 2016 IEEE 8th International Memory Workshop (IMW), Paris, 2016, pp. 1-4.

[52] Shimeng Yu, "Resistive Random Access Memory (RRAM)," in Resistive Random Access Memory (RRAM), Morgan & Claypool, 2016.

[53] Lin, Yu-De et al, "Retention Model of TaO/HfOx and TaO/AlOx RRAM with Self-Rectifying Switch Characteristics." Nanoscale research letters vol. 12,1 (2017): 407.

Weidong Cao Weidong Cao (S'16) received his B.Eng. degree from Northwestern Polytechnical University in 2013, and M.Eng. degree from Tsinghua University in 2016, both in electrical engineering in China. He currently is a Ph.D. candidate in Department of Electrical & Systems Engineering at Washington University in St. Louis. His research interests focus on hardware accelerator, machine learning, in-memory computing, and VLSI Design.



Liu Ke Liu Ke (S'19) is a second-year Ph.D candidate in the Electrical and Systems Engineering department at Washington University in St. Louis. Her research interest lies in design automation and hierarchical modeling of custom machine learning and artificial intelligence accelerators.

Ayan Chakrabarti Ayan Chakrabarti (M'07) is an Assistant Professor in Department of Computer Science and Engineering at Washington University in St. Louis. He received the BTech and MTech degrees in electrical engineering from the Indian Institute of Technology Madras, Chennai, India, in 2006, and the SM and PhD degrees in engineering sciences from Harvard University, Cambridge, MA, in 2008 and 2011, respectively. His research interests are in the fields of computer vision and machine learning, focusing on developing systems that can recover physical reconstructions and semantic descriptions of the world from visual measurements, for applications in robotics, autonomous vehicles, consumer photography, graphics, etc.

Xuan Zhang Dr. Xuan 'Silvia' Zhang (S'08, M'15) is an Assistant Professor in the Preston M. Green Department of Electrical and Systems Engineering at Washington University in St. Louis. She received her B. Eng. degree in Electrical Engineering in 2006 from Tsinghua University in China, and her MS and Ph.D. degree in Electrical and Computer Engineering from Cornell University in 2009 and 2012 respectively. She works across the fields of VLSI, computer architecture, and cyber physical systems and her research interests include adaptive power and resource management for autonomous systems, hardware/software co-design for machine learning and artificial intelligence, and efficient computation and security primitives in analog and mixed-signal domain. Dr. Zhang is the recipient of NSF CAREER Award in 2020, DATE Best Paper Award in 2019, and ISLPED Design Contest Award in 2013, and her work has also been nominated for Best Paper Award at DATE 2019 and DAC 2017.