Materials Horizons

ROYAL SOCIETY OF CHEMISTRY

COMMUNICATION

View Article Online
View Journal | View Issue

Check for updates

Cite this: *Mater. Horiz.*, 2025, 12, 8059

Received 15th January 2025,
Accepted 16th June 2025

DOI: 10.1039/d5mh00086f

rsc.li/materials-horizons

Implementation of ultra-low-power neural networks on quantized and pruned RRAM crossbar arrays†

Hyoseob Kim,‡[a] Kyungho Hong,‡[b] Sungjoon Kim,[c] Woo Young Choi *[b] and Min-Hwi Kim *[ad]

We demonstrate ultra-low-power spiking neural network (SNN) inference on an RRAM crossbar array by applying network lightweight techniques, and predict average power consumption using a highly accurate array-level model. A 24 × 24 crossbar array was fabricated using non-filamentary HT-RRAM, and quantized and pruned weights were transferred to the array. The compact model of HT-RRAM was used as a synaptic device to simulate a crossbar array model of the same scales the fabricated array, and the same network lightweight techniques were applied in the simulation. Both the crossbar array and the array model successfully transferred over 94% of the weights within an error margin of 2 nS, and the SNN inference results over 25 time steps showed highly consistent output currents. With this reliable array model, power consumption during MNIST inference was estimated for arrays with lightweight techniques applied. Based on our experimental results, the power consumption of image inference operations is predicted to be 243 nW with weight quantization only and 222 nW with weight quantization and pruning, across 10 classes. These findings suggest that ultra-low-power operation can be achieved in the RRAM array through the application of lightweight network techniques.

New concepts

In this study, we aimed to predict the power consumption during spiking neural network inference in a crossbar array, employing network lightweight techniques for ultra-low power operation, using a highly accurate model. A 24 × 24 crossbar array was fabricated using non-filamentary resistive random access memory based HT-RRAM, and weights with quantization and pruning applied were transferred to the array. Based on our highly accurate and reliable array circuit model, the power consumption was obtained for image inference in hardware-based neural networks employing low-power resistive memories and network lightweight techniques. Based on our experimental results, the power consumption of image inference operations is predicted to be 243 nW with the weight quantization only and 222 nW with the weight quantization and pruning, across 10 classes. We believe that our research and discoveries will serve as a cornerstone for enhancing the performance of ultra-low-power hardware visual inference systems utilizing next-generation resistive memory and establishing a framework for predicting power consumption.

1. Introduction

Due to rapid growth of artificial intelligence (AI), its applications have expanded into various domains such as character recognition, image generation, and autonomous driving.[1,2] Recently, the growth of large language models (LLMs) has led to the emergence of AI chatbots capable of human-like interactions and contextual understanding, becoming essential parts of people's daily lives.[3,4] To achieve high inference accuracy, these AI models require both huge training and testing on substantial datasets. Traditionally, AI computations are carried out on software-based artificial neural networks (ANNs) running based on the von Neumann architecture, which relies on sequential data transfers between the central processing units (CPUs) and memory. This approach often leads to significant energy consumption and limited efficiency. In an effort to mitigate such bottlenecks, graphics processing units (GPUs) or specialized hardware accelerators, such as neural processing units (NPUs), have been employed to exploit parallel data processing. However, these solutions also cause high power costs and require server-grade hardware for large-scale training and inference tasks. Consequently, there is a growing emphasis on the development of next-generation computing architectures that enable low-power, highly parallel operations,

[a] *Department of Intelligent Semiconductor Engineering, Chung-Ang University, Seoul, Republic of Korea*
[b] *Department of Electrical and Computer Engineering and Inter-university Semiconductor Research Center (ISRC), Seoul National University, Seoul, 08826, Republic of Korea. E-mail: wooyoung@snu.ac.kr*
[c] *Department of AI Semiconductor Engineering, Korea University, Sejong, 30019, Republic of Korea*
[d] *School of Electrical and Electronics Engineering, Chung-Ang University, Seoul, 06974, Republic of Korea. E-mail: minhwi@cau.ac.kr*

† Electronic supplementary information (ESI) available. See DOI: https://doi.org/10.1039/d5mh00086f

‡ H. Kim and K. Hong contributed equally to this work.

thereby addressing the ever-increasing computational demands of contemporary AI systems.

One of the most promising approaches is to emulate the biological neuron and synapse structures in the hardware. By using crossbar array architectures, ANN can rapidly process data in parallel through vector–matrix multiplication (VMM) between stored weights and incoming input signals.[5–8] In particular, the biological brain operates with minimal power consumption by using spike pulses of uniform amplitude and width, enabling event-driven operation based on temporal differences. Therefore, implementing a spiking neural network (SNN) in hardware-based ANN allows for much higher energy efficiency compared to conventional deep neural networks (DNNs) that rely on vector-based input signals.[9,10] Synaptic devices responsible for weight storage may utilize conventional CMOS-based non-volatile memory (NVM).[11,12] However, due to its limitations in speed and scalability, many researchers are focused on next-generation analog memories such as resistive RAM (RRAM), ferroelectric RAM (FRAM), and spin-transfer torque resistive RAM (STT-RAM).[13,14] Among these, RRAM has gained considerable attention for its suitability for array implementation due to its high multi-level cell capability and excellent endurance.[15–19]

Filamentary RRAM, the most widely studied type among oxygen-based RRAM devices, offers advantages in terms of long data retention and high endurance compared to other types. However, notable current variability with the applied set and reset voltages can interfere with precise weight transfer, directly affecting inference accuracy. Additionally, as device scaling is pursued for high-density integration, the forming voltage tends to increase, and the conductivity in the low-resistance state (LRS) after filament formation does not decrease proportionally relative to that of the high-resistance state (HRS). These characteristics can negatively impact conductivity uniformity across the array and interfere with accurate weight transfer when implementing multi-bit functionality.[20–22]

Moreover, several challenges arise during the fabrication of RRAM arrays. Since RRAM performance is highly dependent on the chemical composition of materials, any inconsistency of stoichiometry during deposition can result in imprecise resistance changes in response to voltage. Also, if the thin film is not deposited uniformly over the entire wafer area, inconsistencies in device characteristics may occur between array elements. These issues become even more critical in multilayered RRAM devices. Minor changes in process conditions or incomplete process control can weaken the uniformity of switching behavior across the array. This makes accurate weight transfer difficult and necessitates additional time and cost for optimization. Therefore, it is essential to establish a high-precision array model that can predict the variation of operational characteristics of synaptic devices before array fabrication.

In this study, we used an oxygen vacancy-based non-filamentary RRAM to minimize issues such as non-uniform conductivity and scaling limitations encountered in filamentary RRAM. The 24 × 24 RRAM crossbar array was fabricated using HT-RRAM with HfO_2 and TiO_x as insulating layers, employing it as a synaptic device. To enable ultra-low-power operation, mimicking the efficiency of biological neural networks, a lightweight network was realized by applying quantization and pruning to the pre-trained positive and negative weights, which were then transferred to the array. Subsequently, a 24 × 24 crossbar array was modeled using the PySpice tool, applying the compact model of HT-RRAM to accurately simulate the characteristics of the device under various operational conditions. The weight transfer and MNIST inference results obtained from the physical array and the array model were compared, demonstrating that the array model accurately reflects the behavior of the physical array. A schematic overview of this process is illustrated in Fig. 1. Finally, the average power consumption for 10 classes (from 0 to 9) was calculated using the array model, revealing differences in power consumption when quantization-only and quantization with pruning were applied.

2. Results and discussion

2.1. RRAM device characteristics

For non-filamentary switching operation in oxygen vacancy-based RRAM devices, it is crucial to ensure the uniform generation of oxygen vacancies and oxygen ions across the entire device area. In typical filamentary RRAM, applying set voltage induces the formation of conductive filaments in the resistive switching layer (RSL), connecting the top and bottom electrodes, which results in a sudden increase in current. An effective approach to suppress this behavior involves using a metal oxide with a metal–oxygen bond energy higher than that of the RSL, serving as a barrier layer to inhibit the formation of conductive filaments. In this work, we fabricated HT-RRAM devices using TiO_x as the RSL and HfO_2 as the barrier layer. As shown in Fig. S1 (ESI†), the graph illustrates the bond energies between various metals and oxygen.[23] According to this data, the Hf–O bond energy (791 kJ mol^{-1}) is higher than the Ti–O bond energy (662 kJ mol^{-1}), suggesting that the HfO_2 barrier layer can effectively inhibit the formation of conductive filaments within the RSL, even when the set voltage is applied.

Fig. 2a illustrates the fabrication process of the HT-RRAM device. The process was conducted on a 4-inch heavily doped p-type silicon wafer, with a 300 nm silicon oxide layer formed through a wet oxidation process. First, the bottom electrode (BE) was deposited using an electron-beam evaporation system, where 10 nm of Ti and 50 nm of Pt were deposited. Patterning was carried out through a lift-off process. Next, TiO_x was deposited to a thickness of 9 nm *via* oxygen reactive sputtering which serves as the RSL. To analyze the electrical properties as a function of the oxygen concentration in the RSL, TiO_x was grown at three oxidation conditions (0.3, 0.7, and 1.1 sccm). In the third step, a barrier layer of HfO_2 was deposited to a thickness of 4.5 nm using atomic layer deposition (ALD). In the fourth step, the top electrode (TE) was deposited using the electron-beam evaporation system, with 10 nm of Ti and 50 nm of Pt. Patterning was carried out through the lift-off process. Finally, residual oxide was etched to remove the remaining

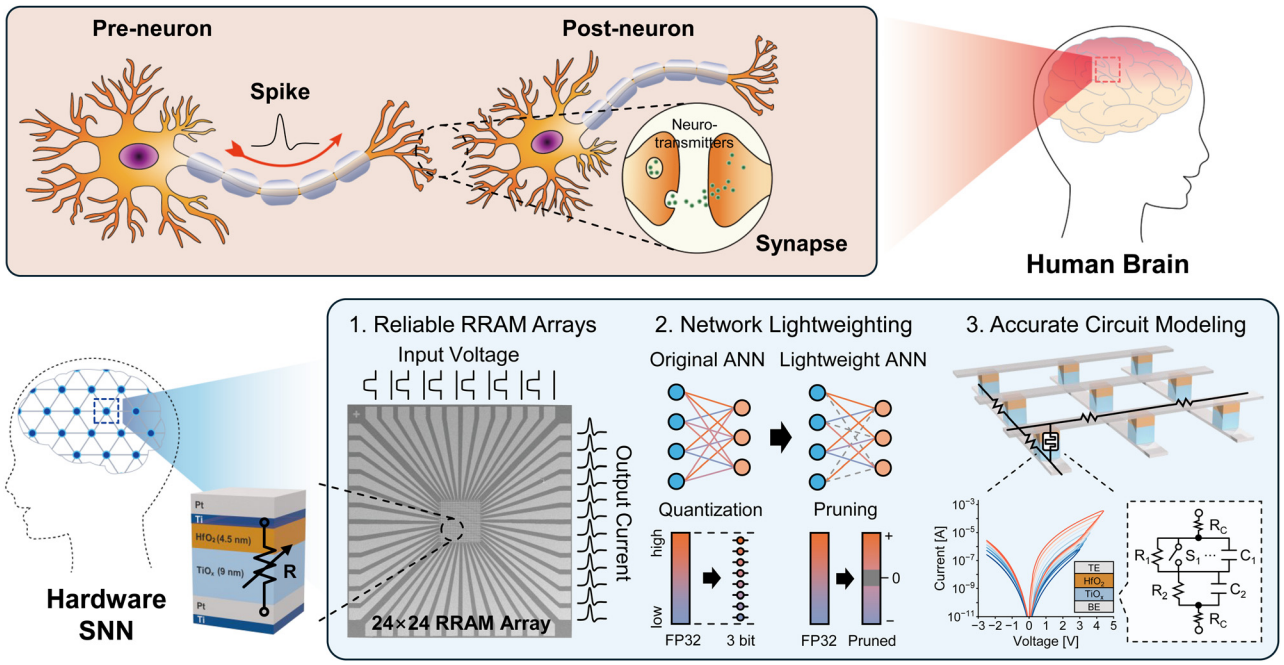

Fig. 1 Schematic diagram of the inference system of a biological neural network and hardware-based spiking neural network implemented with RRAM crossbar array. Low power consumption is achieved by applying lightweight techniques to the crossbar array, which is also implemented as a reliable simulation model to estimate power consumption.

oxides and to ensure the stable performance of the device. Additionally, a reference filamentary RRAM device, comprising a Pt/Ti/HfO_2/Pt/Ti stack, was fabricated without the deposition of TiO_x. Fig. 2b presents the high-resolution transmission electron microscopy (HR-TEM) image and energy dispersive X-ray spectroscopy (EDS) line scanning of the fabricated HT-RRAM device. The HR-TEM image confirms that all layers were stacked as intended, indicating the successful operation of the fabrication process. For a more detailed analysis, the atomic percent obtained from EDS line scanning clearly shows the presence of approximately 4.5 nm of HfO_2 between Ti and TiO_x, while TiO_x was confirmed to have a thickness of around 9 nm. Additionally, Fig. 2c presents the 3D schematics of both the HT-RRAM and the reference RRAM. To analyze the non-filamentary switching behavior of the HT-RRAM, we compared the reference and HT-RRAM devices after performing a DC sweep. As shown in Fig. 3a, the reference RRAM device showed an abrupt set at 1 V and a gradual reset at −1 V after the forming process, which are characteristic of filamentary RRAM behavior. This behavior is consistent with that of the TiO_x-based RRAM, as shown in Table S1 (ESI†). For the HT-RRAM devices with oxygen composition of 0.3, 0.7, and 1.1 sccm, the *I–V* characteristics were measured under the same voltage conditions applied to the top electrode, up to hard breakdown. The corresponding results are presented in Fig. 3b–d. In contrast to the reference RRAM device, the HT-RRAM exhibited a gradual increase in current upon the application of the set voltage under all three oxygen partial pressure conditions. This behavior confirms that the HfO_2 layer effectively suppresses the formation of conductive filaments within the TiO_x layer. While several previously reported RRAM devices share a similar HfO_2 and TiO_x bilayer structure, our device uniquely operates without a compliance current or forming voltage, due to its non-filamentary switching mechanism. Comparative details are summarized in Table S2 (ESI†).

Moreover, under all three oxygen partial pressure conditions, the set voltage gradually increased in approximately 0.2 V intervals, ranging from 3 V to just before hard breakdown, enabling multi-level conductance control. However, hard breakdown occurred at an average 4.4 V and permanently altered the conductance in the LRS state, making it essential to avoid such conditions during switching operations. Therefore, the voltage range for safe multi-level conductance switching is limited to 3 V to 4.3 V.

To compare the electrical characteristics of HT-RRAM under three oxygen compositions, each *I–V* curve obtained with a 4 V applied voltage was overlaid, as shown in Fig. 3e. At 1 V, the 1.1 sccm condition exhibited the widest current window, which can be attributed to its higher maximum current under the same set voltage. These results indicate that higher oxygen partial pressure within the RSL increases the probability of oxygen vacancy generation, resulting in a relatively higher conductivity. Such characteristics lead to an expanded current window range, suggesting that oxygen partial pressure directly impacts the size of the current window in the switching layer.

Lastly, the scalability effects of the HT-RRAM device were assessed by measuring cells with four distinct sizes: 2.5 μm × 2.5 μm, 5 μm × 5 μm, 10 μm × 10 μm, and 20 μm × 20 μm. This approach allowed for a comparative analysis of the devices performance across varying cell areas, providing insights into

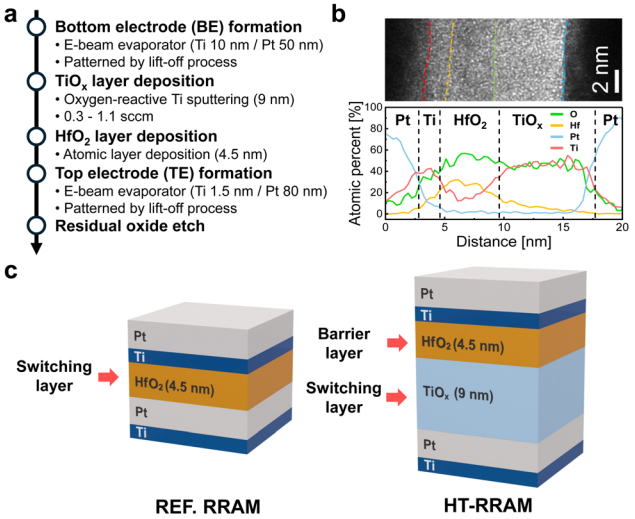

Fig. 2 (a) Fabrication process flow of the HT-RRAM. (b) TEM image and EDS line scan of HT-RRAM. (c) Comparison of stack structure and layer thickness between the HT-RRAM and reference RRAM.

how scaling impacts the electrical characteristics of non-filamentary RRAM. All devices were initially set to the HRS state, and a 4 V bias was applied to the TE to switch them to the LRS state. Under the same bias conditions, larger cell areas exhibited higher current levels. Fig. 3f presents box charts showing the current densities of LRS and HRS states measured from 10 devices per each cell size. The current density was calculated by dividing the current measured at a 1 V read voltage by the respective cell area. The box charts reveal that current density is nearly unaffected by cell size, demonstrating a key characteristic of non-filamentary RRAM devices. Unlike filamentary RRAM devices, where current flows through highly localized conductive filaments, non-filamentary devices exhibit uniform current flow across the entire cell area, resulting in consistent current density. This uniformity highlights a significant advantage for high-density integration: even with a reduction in device size, the electrical characteristics remain consistent, making non-filamentary devices ideal for array fabrication and miniaturization.

Synaptic devices intended for application in crossbar arrays benefit from enhanced multi-level functionality when the current window is wider within the same switching voltage range. To meet these requirements, we aimed to implement an array using the 1.1 sccm HT-RRAM, which demonstrated a larger current window under these conditions. Initially, the multi-level characteristics of the 1.1 sccm HT-RRAM were evaluated. Based on the switching voltages identified in the *I–V* characteristics, ISPP (incremental step pulse programming) was applied within a range that avoids hard breakdown. Set pulses, starting at 3 V and increasing in 25 mV increments up to 4.3 V, and reset pulses, starting at −1 V and decreasing in 50 mV increments down to −2.5 V, were sequentially applied to the device in its initial state. Between each pulse, a read pulse of 1 V was applied, and the resulting current values were recorded and presented in Fig. 4a. The width of all pulses was set to 30 ms.

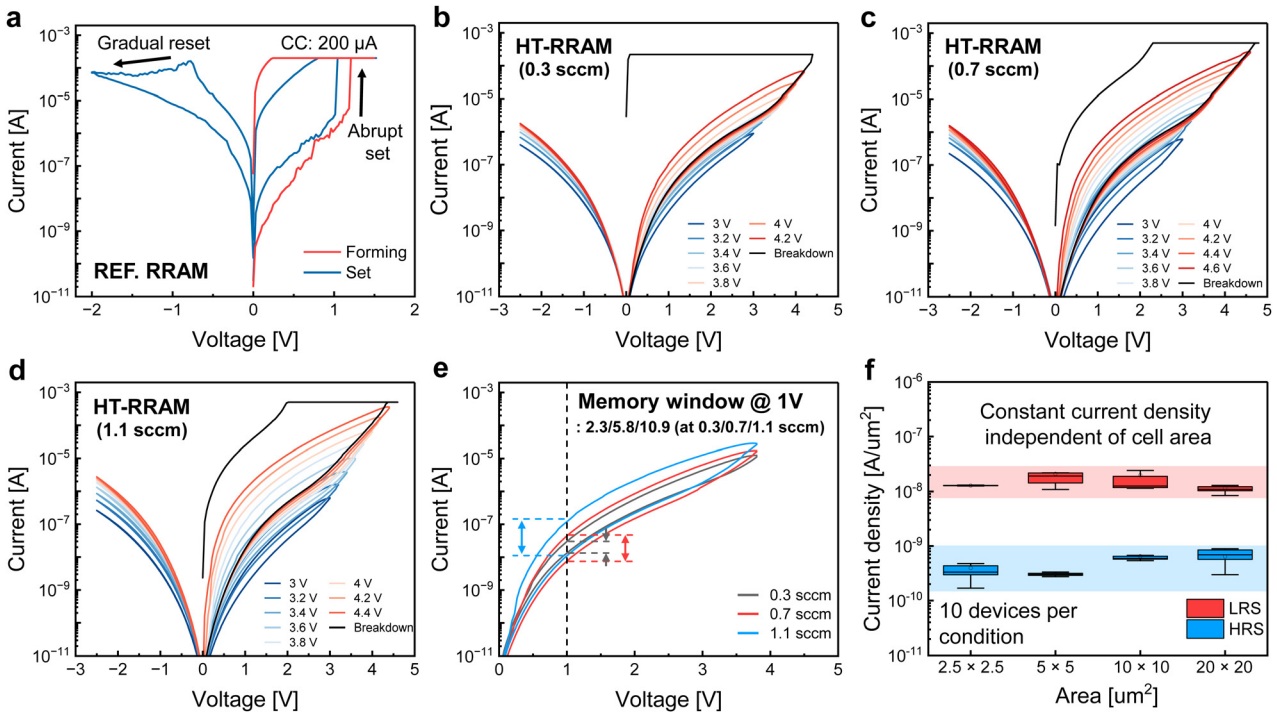

Fig. 3 (a) *I–V* characteristics of the reference RRAM with a single HfO_2 layer. (b)–(d) *I–V* characteristics of HT-RRAM based on the oxygen content in the TiO_x layer. (e) Graph comparison at 4 V set under oxygen composition. As the oxygen composition increases, the current window at 1 V read shows an increasing trend. (f) Box chart representing current density in LRS and HRS for various cell sizes. Measurements were conducted on 10 devices per condition, showing consistent current density regardless of cell size.

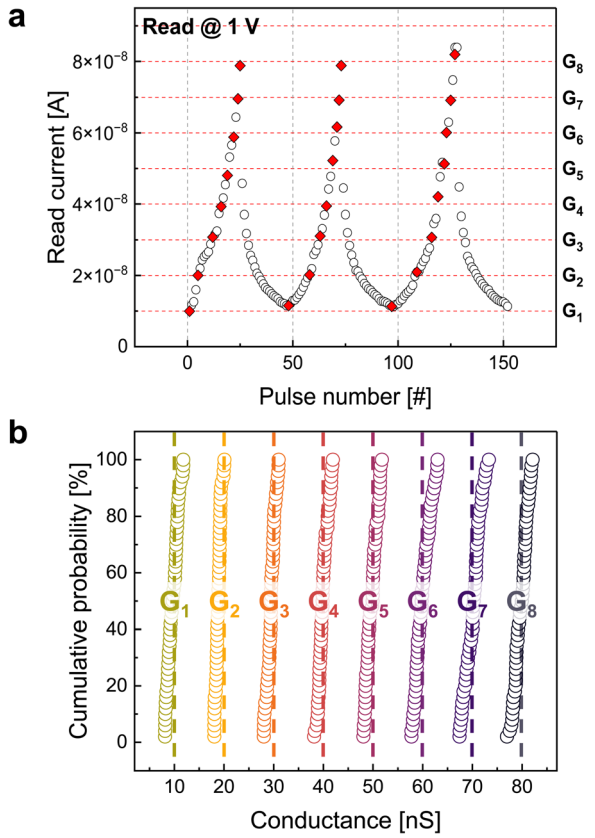

Fig. 4 (a) ISPP result of single HT-RRAM over three cycles. (b) Cumulative distribution of 3-bit target states (G_1 to G_8) in HT-RRAM. About 94% of values were tuned within a 2 nS error margin and the remaining values were tuned within a 4 nS error margin.

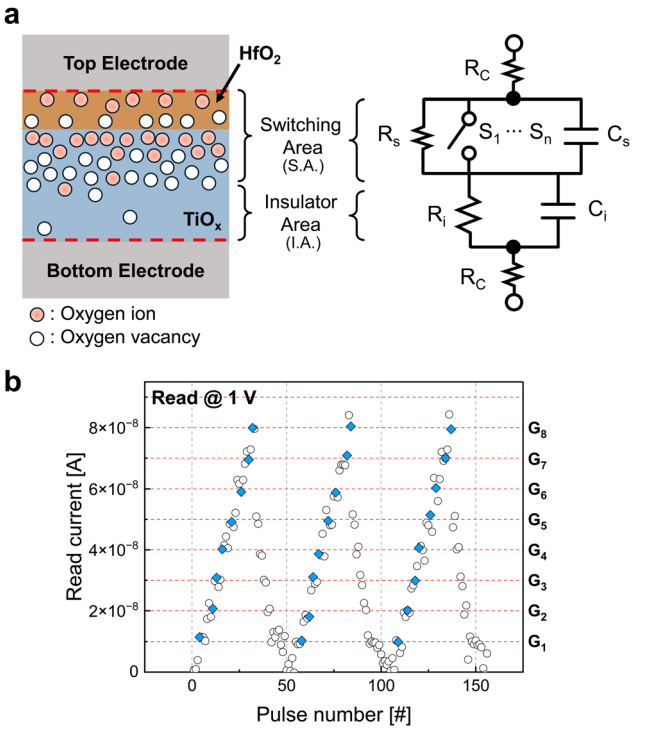

Fig. 5 (a) Designed compact model of HT-RRAM that was implemented in PySpice. By connecting multiple voltage-controlled switches in parallel, gradual set behavior is induced. (b) ISPP simulation result of the compact model over three cycles.

The results of three cycles revealed a repeatable current range of approximately 0 nA to 80 nA, demonstrating the potential for 3-bit weight tuning (G_1 to G_8) with 10 nA intervals at 1 V. In addition, the gradual current increase observed in the long-term potentiation (LTP) region suggests that it is suitable for reaching specific target states, whereas the abrupt current drop in the long-term depression (LTD) region indicates that it should be used solely for resetting the conductance state. Based on these findings Fig. 4b presents the cumulative probability of each tuned state using the incremental step pulse verification algorithm (ISPVA). Conductance was also measured at a 1 V read voltage, and the results include data from 50 devices for each state. Notably, 94% of the total weights were tuned with an error margin of 2 nS, while the remaining values were tuned within a 4 nS error margin. These error margins are well within the acceptable range, given the target weight interval of 10 nS. This indicates that the device can be quantized and programmed as a 3-bit weight in a 24 × 24 array configuration.

2.2. RRAM device compact modeling for the array model

Based on the measured characteristics of the actual device, a compact model was developed. This allows efficient analysis that would be challenging to perform *via* measurements, such as repeated execution of VMM operations, monitoring of individual current paths, and systematic evaluation of power consumption. Both compact modeling of the device and array modeling were implemented with PySpice. PySpice enables circuit design through Python scripts without a graphical user interface (GUI), making it significantly simpler and faster to model high-density crossbar arrays with repetitive structures compared to conventional SPICE tools. The absence of graphic processing further reduces simulation time. In HT-RRAM, the HfO_2 layer serves as a barrier that prevents the formation of conductive filaments in the TiO_x layer between the two electrodes. This mechanism enables the generation of oxygen ions and vacancies throughout the insulating region, and the resulting variation in their concentration leads to changes in conductivity. As illustrated in Fig. 5a, the compact model of HT-RRAM employs voltage-controlled switches connected in parallel, allowing the insulating region (switching area) to gradually adjust its conductivity based on the applied voltage. Each switch modulates its resistance according to the threshold voltage (V_T) and the hysteresis voltage (V_H), and connecting multiple switches in parallel helps prevent an abrupt set of devices. In the lower part of the insulator (insulator area) underneath the switching area, fewer oxygen ions and vacancies are generated and they do not significantly affect the abrupt resistance changes, so an additional resistor is placed here. Furthermore, because the current in RRAM changes exponentially during voltage sweeps, resistors in both the switching area and the insulator area are given nonlinear characteristics. As both areas are intrinsically insulators, resistance and capacitance coexist, and capacitance is therefore included in the model. Finally, to implement the effect of contact

Table 1 Circuit element, value, and description of HT-RRAM compact modeling

Element	Value	Description
R_s	$51 \times 10^5 \cdot \exp[-(\mathrm{abs}(V_{Area1}) - 2.52)/0.35]\ \Omega$	Non-linear switch resistance
R_i	$87 \times 10^2 \cdot \exp[-(\mathrm{abs}(V_{Area2}) - 4.52)/0.485]\ \Omega$	Non-linear insulator resistance
R_C	100 Ω	Contact resistance
C_s	0.1 fF	Switch Capacitance
C_i	0.1 fF	Insulator Capacitance
S_1–S_n	V_T (threshold voltage): 0.85–1.55 V V_H (hysteresis voltage): 2.4 V R_{ON} (on resistance): 30 kΩ–62 MΩ R_{OFF} (off resistance): 10^{12} Ω	Switch resistances (HRS to LRS)

resistance, additional resistors are placed at both the top and bottom of the circuit. However, since these parameters (R_c, C_s, C_i) are not directly relevant to this study, arbitrary values were assigned to ensure that they do not affect the experimental results. Table 1 summarizes the circuit elements, values, and their descriptions used in the compact model. Finally, to replicate the variability in current observed in actual devices due to stoichiometric differences, Gaussian random values were added *via* Python coding to allow conductance to fluctuate within a defined range. The variables set in the compact model can be customized at any time to reflect the device's characteristics and variability.

Results from DC sweep simulation demonstrate that the compact model exhibited a gradual set behavior during a voltage sweep from 0 V to 4.2 V, while reset behavior occurred during a sweep down to −1.6 V. To further evaluate the feasibility of implementing multi-level capability, an ISPP simulation was conducted. In this simulation, set pulses were sequentially applied starting at 3 V and increasing in 25 mV increments up to 4.2 V. Subsequently, reset pulses were applied, starting at −1 V and decreasing in 50 mV increments down to −1.6 V, effectively replicating the actual measurement scheme. Fig. 5b presents the current values read at a 1 V pulse between each programming pulse. The pulse width was set to 2 ms. Over three cycles, the current values were observed to consistently range from approximately 0 nA to 80 nA. This behavior closely matches that of the actual device, confirming its potential for 3-bit weight tuning (G_1 to G_8) with 10 nA intervals at 1 V. These results indicate that the compact model was designed successfully to operate in a way closely aligned with the behavior of the actual device.

2.3. 24 × 24 RRAM crossbar array and SNN implementation

Fig. 6 illustrates the overall SNN structure used to infer a specific digit. This network consists of a two-layer architecture with dimensions of 784 × 24 × 10, designed to process the modified national institute of standards and technology database (MNIST) dataset and output one of ten possible results. Specifically, one of the 2000 test digit images (from 0 to 9) is fed into the input layer. The 28 × 28 pixels of the image are converted into spike rates based on pixel brightness and transmitted as input signals.

The signals and the weights stored in the first layer perform VMM operations to produce currents, which are integrated and fired by neurons to generate spikes for the second layer. The second layer, a fully connected 24 × 10 network, outputs currents through VMM operations involving the spikes from the first layer and its stored weights. This layer is implemented using an HT-RRAM based crossbar array and an array model using PySpice, as shown in Fig. 6b and c. The array model, based on the compact model, was designed to incorporate line resistance. Further details are provided in Fig. S2 (ESI†).

In both cases, the system is designed to accept spike trains over 25 time steps and perform VMM operations at each timestep to produce current outputs. To represent both positive (I_{G^+}) and negative (I_{G^-}) weight values, the network requires differential pairs of 24 × 10 crossbar arrays. As illustrated in Fig. 6(c), a single 24 × 24 crossbar array is used, where positive weights range from $I_{G^+}(1, 1)$ to $I_{G^+}(10, 24)$, occupying rows 1 to 10 and columns 1 to 24, while negative weights range from $I_{G^-}(1,1)$ to $I_{G^-}(10,24)$, occupying rows 11 to 20 and columns 1 to 24.[24] Input spikes are applied to 24 wordlines (input nodes), and the VMM operations direct the resulting outputs to 20 bitlines (output nodes). The outputs for positive and negative weights are subtracted, and the current sum determines the correct digit based on the output node with the highest current. The overall classification accuracy is calculated using the same method. The array model simulations were conducted under identical conditions.

Therefore, we performed software-based pre-training using Python simulations (ANN with ReLU activation) to ensure that all layers include both positive and negative weights. This off-chip learning approach allows only the inference operation to be executed on the physical array, making it advantageous for low-power operation in resource-limited edge devices. Two types of pre-training were performed: 3-bit quantization (QT) and 3-bit quantization & 40% pruning (QT & PR). Fig. 6d shows the weight histograms for all layers under FP32 and QT conditions, with QT presented separately for each layer. Meanwhile, FP32 weights are distributed across a wide range, and QT restricts weights to specific discrete values. For QT & PR, a pruning ratio of 40% was applied to the FP32 weights using a pruning method, and quantization was performed afterwards.[25–27] Fig. 6e illustrates the weight histograms for the second layer after applying QT and QT & PR. In the case of QT & PR, weights near −0.25 and 0.25 disappear due to pruning. All pre-trained weights were designed to maintain an accuracy of 90% during ANN inference, as shown in Fig. 6f, which compares the weight accuracy for each case.

Subsequently, the pre-trained weights were optimized for HT-RRAM characteristics by mapping them to conductance values ranging from 10 nS to 80 nS with 10 nS intervals. SNN simulations were performed to verify that these optimized

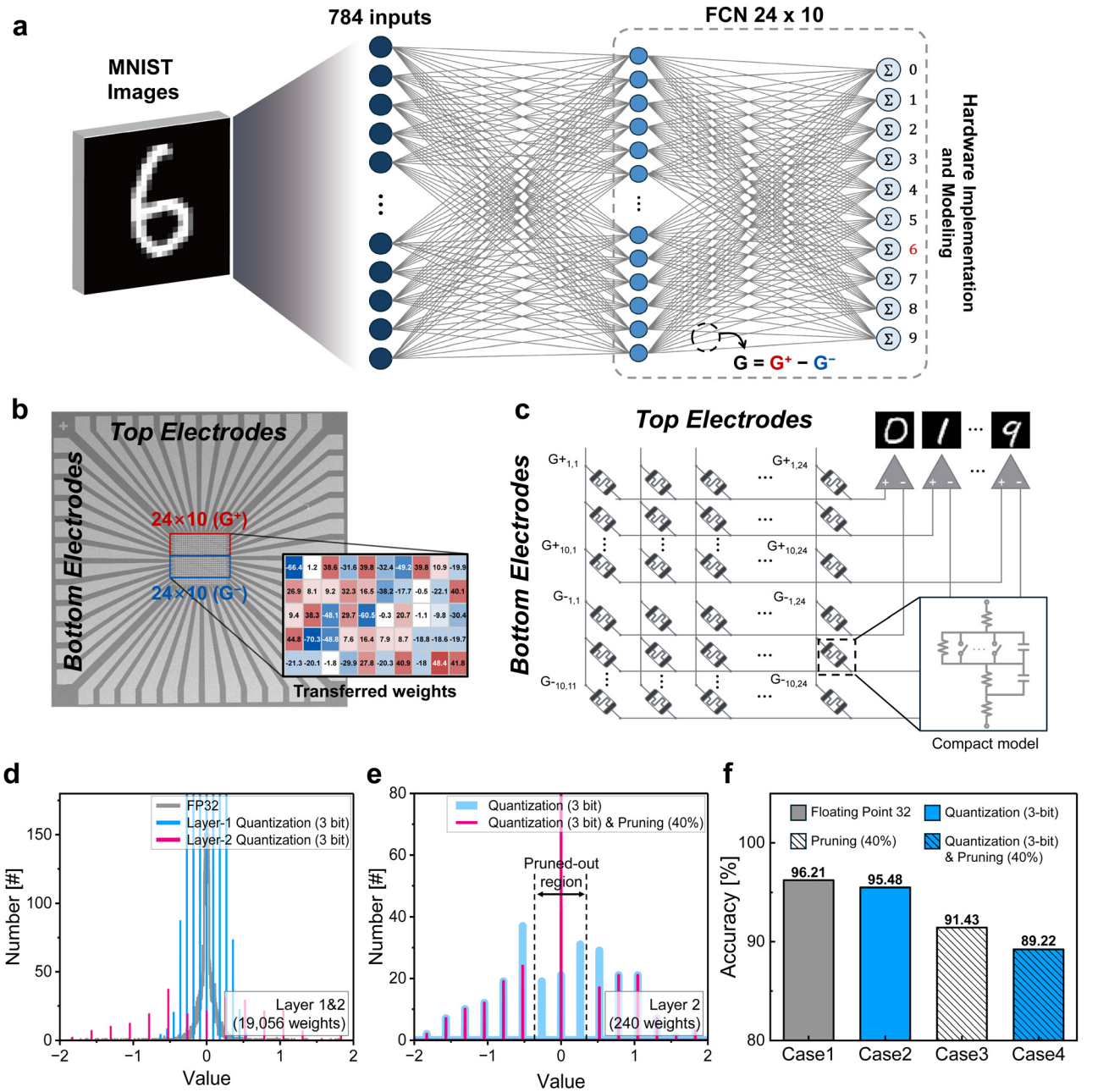

Fig. 6 (a) Schematic of the 784 × 24 × 10 SNN structure. The spikes of a 28 × 28 pixel image, converted *via* rate encoding, are input into the first layer at each timestep. The spikes generated through VMM are then transmitted to the second layer. The fully connected 24 × 10 network was implemented using both hardware and modeling. (b) The 24 × 24 crossbar array fabricated using HT-RRAM, where the weights are stored in the cells of the array. (c) 24 × 24 crossbar array model with compact modeling of the designed HT-RRAM. To implement both positive and negative weights, the model has 20 outputs as in the physical array. Each digit from 0 to 9 is identified through current differences. (d) Weight histograms for all layers show 32-bit floating point weights distributed widely, whereas quantization confines them to discrete values. (e) The weight histograms of the second layer after quantization and quantization & pruning show that from −0.25 to 0.25 were eliminated through pruning. (f) The pre-trained results extracted weights that ensured over 90% accuracy even in the quantization & pruning case.

weights maintained their accuracy. Furthermore, the SNN simulation converted the ANN inputs of each layer spike train, facilitating the ANN to SNN conversion process.

Following the conversion, different methods were employed to transfer the optimized weights to the fabricated HT-RRAM array and the array model. The fabricated RRAM array focused on precise transfer, whereas the array model was designed to minimize simulation time. As shown in Fig. 7a, the RRAM crossbar array used the half-bias method to control the conductance of individual cells.[28] In this method, half of the target voltage with opposite polarity was applied to the TE and bottom electrode (BE) using an SMU pulse. A rectangular pulse with a

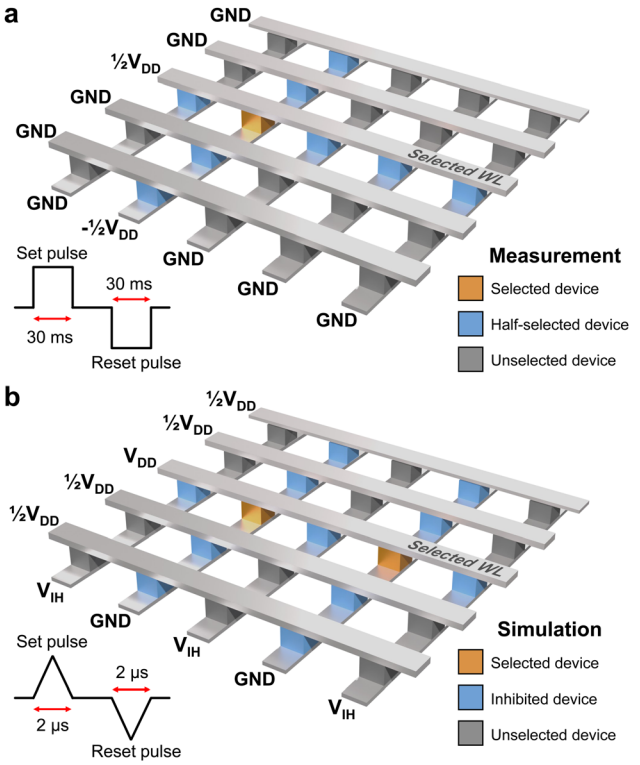

Fig. 7 (a) Schematic of the weight transfer method in an RRAM crossbar array. The half-bias method was employed using rectangular pulses with a width of 30 ms. (b) Schematic of the weight transfer method in an RRAM crossbar array model. The fast weight transfer method was applied using triangular pulses with a 2 μs width.

20 ms width was used to adjust the conductance, and a 1 V read pulse was applied to measure the adjusted conductance. If the measured value exceeded the allowable tolerance from the target state, set and reset pulses were applied iteratively. The pulse voltage was adjusted based on measured error to quickly reach the target state. Furthermore, the voltage applied to half-selected devices was automatically limited to half of the target voltage, preventing unintended conductance changes and enabling accurate weight transfer.

Furthermore, Kim *et al.* proposed a fast weight transfer method designed for real-time online learning in RRAM-based neuromorphic systems, demonstrating significant improvements in the efficiency of weight update operations.[29] In this study, the same method was applied to minimize the simulation time for weight transfer. Fig. 7b illustrates the fast weight transfer method. A triangular pulse with a 2 μs width was applied to the selected wordline at the target voltage, while the bitline was driven with either ground (GND) or an inhibit voltage (V_{IH}) to control writing or inhibition. For inhibited wordlines, half of the target voltage was applied to effectively suppress write disturbances in unselected cells. The key advantage is that devices sharing the same weight level on a wordline are transferred simultaneously, which not only ensures reliable data writing but also reduces the weight transfer simulation time by half compared to the half-bias method.

Using the above methods, we completed the weight transfer and evaluated the accuracy of the transferred weights. Fig. 8

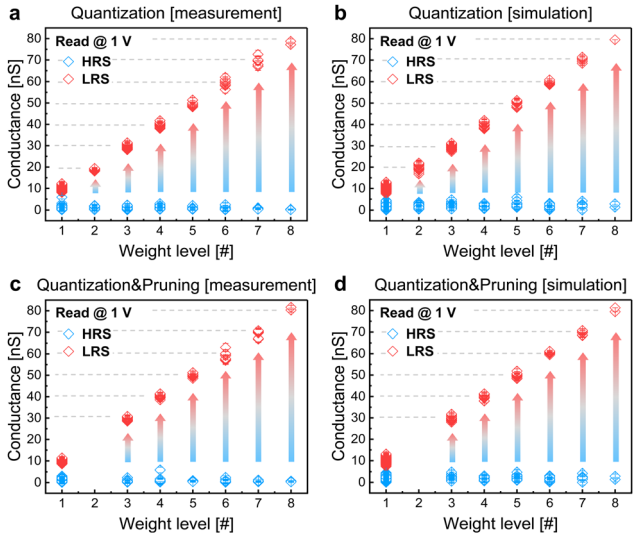

Fig. 8 Comparison of conductance changes according to weight levels in the (a) quantization applied crossbar array and (b) array model. Comparison of conductance changes according to weight levels in the (c) quantization & pruning applied crossbar array and (d) array model. In both network lightweight methods, each measurement and simulation shows that 240 devices are transferred at 10 nS intervals from 10 nS to 80 nS under a 1 V read condition.

illustrates the change in conductance for 240 devices transferred from the RRAM crossbar array and the array model, classified by weight levels. In both the measured and simulated QT and QT & PR cases, the distinction between LRS and HRS is clearly observed. HRS conductance values are consistently distributed near 0 nS regardless of the weight level, while LRS values show conductance changes corresponding to their

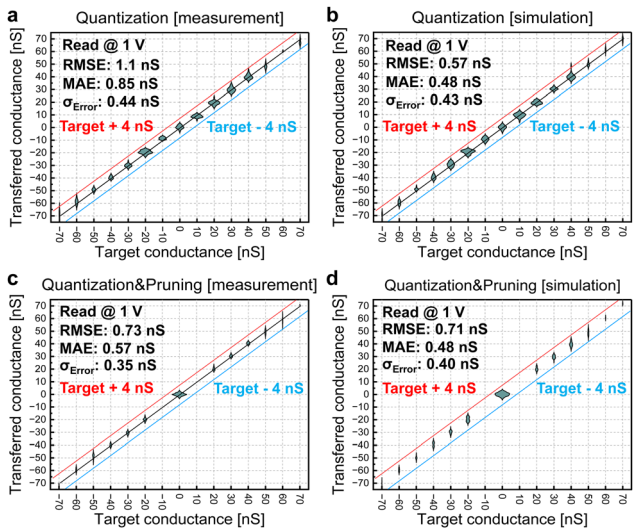

Fig. 9 (a) and (b) Comparison of violin plots for crossbar arrays and array models implementing both positive and negative weights after quantization. (c) and (d) Comparison of violin plots for crossbar arrays and array models implementing both positive and negative weights after quantization & pruning. The measurement and simulation results exhibit a high degree of similarity.

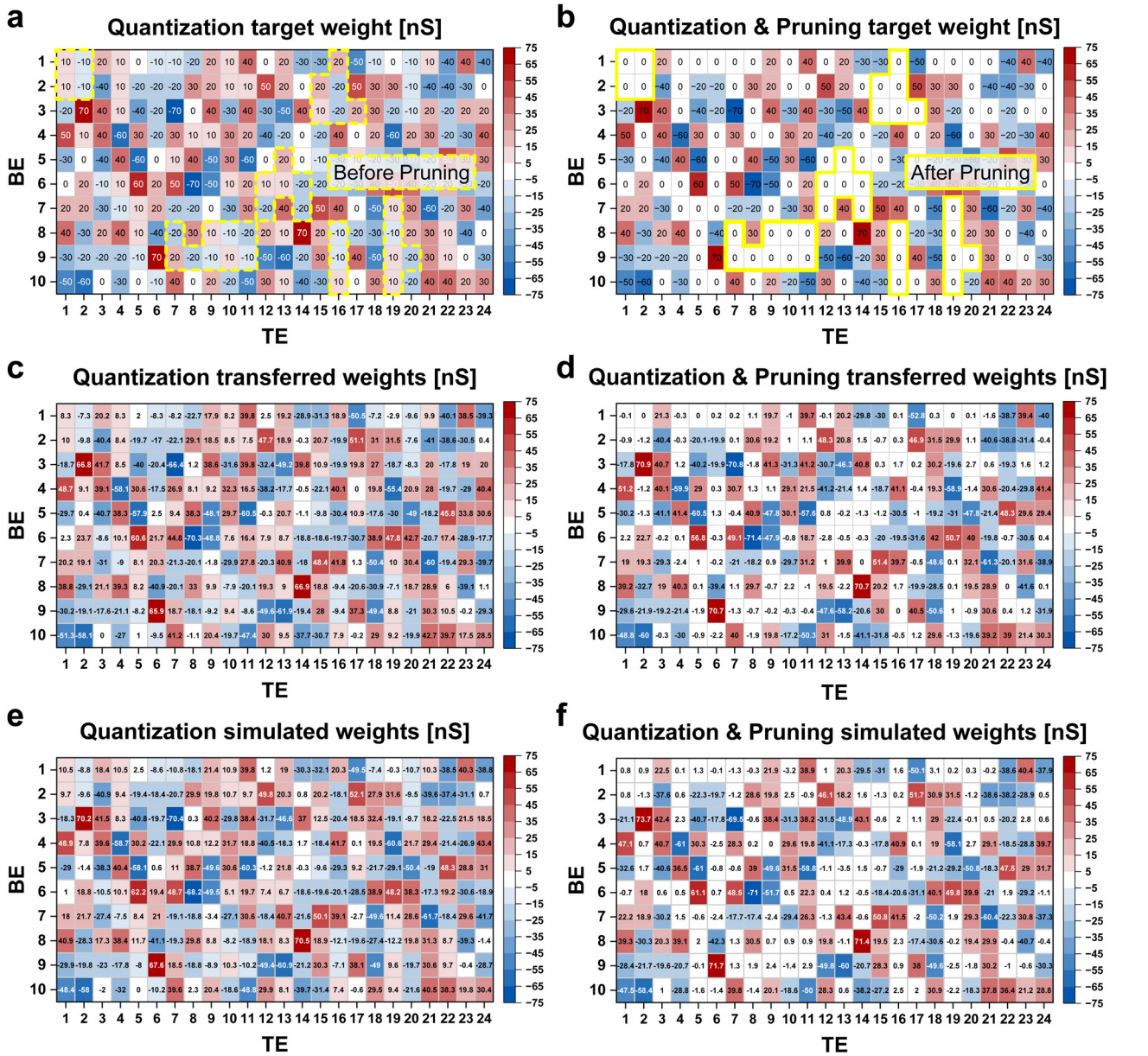

Fig. 10 (a) and (b) Target weight map applied with quantization and quantization & pruning. (c) and (d) Transferred weight map applied with quantization and quantization & pruning. (e) and (f) Simulated weight map applied with quantization and quantization & pruning. Both the transferred and simulated weight maps exhibit values closely aligned with the target weight map.

respective weight levels within the allowable variance. Notably, in the QT & PR case, the 20 nS weight level is absent due to pruning. This also indicates that our HT-RRAM operates within a smaller conductance range compared to RRAM arrays reported over the past five years, highlighting its advantage in achieving ultra-low-power operation (see Table S3, ESI†). Fig. 9 presents a violin plot that compares the measured and simulated results for both positive and negative weights. The root mean square error (RMSE) and mean absolute error (MAE) are 1.1 nS and 0.85 nS for the measured QT results, and 0.57 nS and 0.48 nS for the QT & PR results. This indicates that, despite a few outliers, most of the weights were transferred with an allowable error margin. Also, the RMSE and MAE for the measured QT & PR results are 0.57 nS and 0.48 nS, respectively. The simulation results also closely matched the measured data, confirming the strong alignment of the array model.

Fig. 10 illustrates the weight maps of the target, the RRAM crossbar array, and the array model after weight transfer under both QT and QT & PR cases. The weight map represents the calculated values of I_{G^+}–I_{G^-}, and Fig. S3 (ESI†) displays the detailed weight maps for I_{G^+} and I_{G^-}. These results confirm that we conclude that the weights were successfully transferred to the RRAM crossbar array. Moreover, the results demonstrated that a reliable array model has been established, closely replicating not only the measurement data, but also its associated error margins.

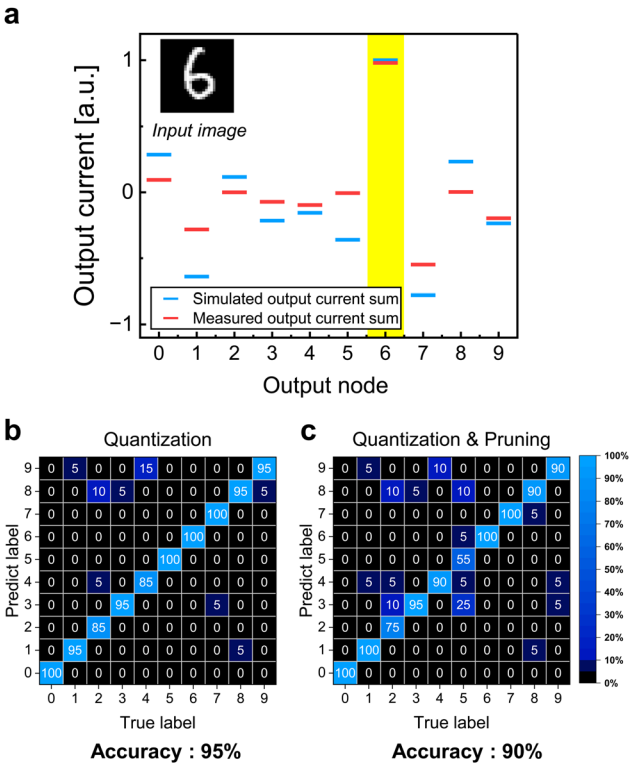

Fig. 11 (a) Comparison of current output results between the RRAM crossbar array and array model when the same image (number 6) is input. The current output at each node from the measurements and simulations is found to be highly similar. Confusion matrix generated from inference results of 20 samples per class for both (b) quantization and (c) quantization & pruning using the array model.

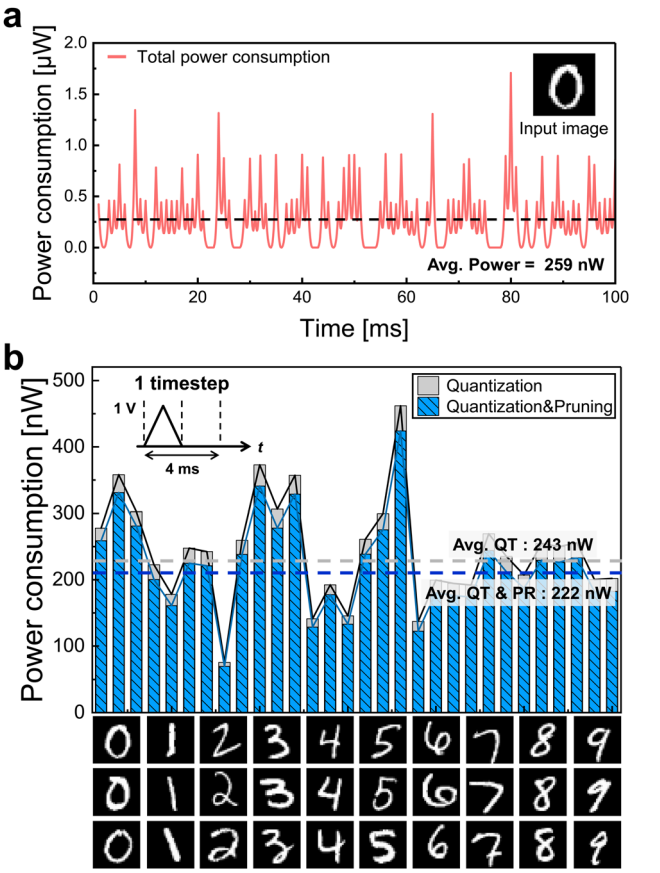

Fig. 12 (a) Total power consumption of the array and average power consumption ($P_{AVG.array}$) over time during inference. (b) $P_{AVG.array}$ compares QT and QT & PR during SNN inference. Power consumption for inference on three random images from a single class, demonstrating that QT and QT & PR consistently consume less power than QT, regardless of the image class.

After completion of the weight transfer process, VMM operations were performed using test image 6. Fig. 11a shows the current values of the output nodes for the RRAM crossbar array measurement and the array model simulation. In both cases, the highest current output was observed for digit 6, and the current outputs from other output nodes exhibited significantly similar trends. This implies that the array model not only emulates weight transfer with high fidelity, but also performs SNN inference in a manner aligned with the physical array. Notably, during inference, all bitlines in both the simulated array model and the physical array are grounded, thereby preventing the occurrence of sneak paths. More details are illustrated in Fig. S7 (ESI†). Additionally, to further validate the VMM operations of the array model, classification tests were conducted using 200 random MNIST test images (20 images per digit) across all digits. The confusion matrix in Fig. 11b and c presents the accuracy of inference for each digit under the conditions QT and QT & PR. QT achieved an average inference accuracy of 95% in all classes, while QT & PR maintained an average accuracy of 90% despite pruning. Interestingly, these results are highly consistent with the pre-training accuracy obtained through Python simulations, as shown in Fig. 6f. Thus, the array model implemented in this study can be considered reliable.

As illustrated in Fig. S6 (ESI†), during the inference process, the current measured on a single wordline in the array is equal to the sum of the currents flowing out from the 20 connected bitlines. Consequently, the power consumed by this wordline over time can be considered the sum of the power consumed by the 20 connected RRAM devices, as presented in eqn (1).[30,31]

$$P = VI = V^2/R, \quad \sum_{j=1}^{20} P_{ij}(t) \tag{1}$$

As presented in eqn (2), integrating the time-dependent power consumption of a wordline over the entire inference period and then dividing by that period calculates the average power consumption ($P_{AVG.i}$) of a single wordline. Furthermore, because the array implemented in this study contains 24 wordlines, the total sum of the $P_{AVG.i}$ values of all wordlines calculates the average power consumption of the array ($P_{AVG.array}$), as shown in eqn (3).

$$\frac{1}{T}\int_0^T \sum_{j=1}^{20} P_{ij}(t)\mathrm{d}t = P_{\mathrm{AVG.i}} \tag{2}$$

$$\sum_{i=1}^{24} P_{\mathrm{AVG.i}} = P_{\mathrm{AVG.array}} \tag{3}$$

The time-dependent power consumption of each individual wordline in the array can be visualized, as shown in Fig. S4

(ESI†). The figure also demonstrates that more frequent voltage application leads to increased power consumption. Using this method, we finally computed the power consumption during SNN inference. Fig. 12a shows the total power consumption over the inference time in the array and the corresponding $P_{AVG.array}$ for the case when the input digit is 0. Fig. 12b compares the $P_{AVG.array}$ values during inference on three randomly selected MNIST images for each digit (class) under the QT and QT & PR cases. The total inference time was 25 timesteps, and all images were correctly classified without errors in the results. For all images, QT & PR consumed less power than QT. On average, the $P_{AVG.array}$ values for QT and QT & PR were 243 nW and 22 nW, respectively, which corresponded to approximately 8.6% reduction in power consumption with QT & PR. Although increasing the current pruning ratio beyond 40% could further decrease the power consumption, it may also lead to a decrease in inference accuracy, as illustrated in Fig. S5 (ESI†), which shows trade-off between the pruning ratio and the accuracy of inference. Therefore, determining the optimal pruning ratio for applications that require low power consumption necessitates a careful balance between accuracy and power consumption.

3. Conclusion

In this study, we fabricated a non-filamentary HT-RRAM using HfO_2 as a barrier layer and TiO_x as a resistive switching layer to prevent abrupt current increases and achieve precise multi-level conductivity. DC measurements of individual devices demonstrated the elimination of abrupt set behavior while maintaining uniform current levels, even as the device area was reduced. Through ISPP and ISPVA, we confirmed that the device could be tuned into 8 discrete states (3-bit), and that 94% of 400 single devices were tuned within an error margin of 2 nS. Furthermore, the HT-RRAM compact model and crossbar array were implemented using the PySpice tool. The results of the ISPP simulation confirmed the potential to represent 3-bit states, similar to the actual HT-RRAM device.

For ultra-low-power consumption, we applied two types of lightweight network: QT and QT & PR to pre-trained FP32 weights. The weights were then successfully transferred to the fabricated 24 × 24 HT-RRAM crossbar array. Through simulations, a successful transfer of the lightweight networks showed a weight map that was highly similar to that of the physical crossbar array, including error margins. The SNN inference results for digit 6 using both the RRAM crossbar array and the array model revealed that the highest current output occurred at the same output node, with closely matching current patterns across all nodes. This confirms that we have implemented a highly reliable array model. Simulation-based verification of VMM operations demonstrated that QT and QT & PR achieved average accuracies of 95% and 90%, respectively, on 200 random MNIST images. Finally, average power consumption calculated during inference (25 timesteps) showed that the two types of lightweight networks consumed 243 nW and 222 nW, respectively, confirming that QT & PR consumed approximately 8.6% less power than QT.

These results suggest that ultra-low-power SNN inference can be achieved not only through network quantization but also by applying pruning techniques. Moreover, the array model developed in this study demonstrated its ability not only to predict operational characteristics but also to accurately estimate error margins when implementing HT-RRAM devices at the array scale. Additionally, the modeling suggests the potential to implement various RRAM devices at the array level.

4. Experimental section

4.1. Electrical characterization

The electrical characteristics of the HT-RRAM devices were evaluated using a Keysight 4156C, a semiconductor parameter analyzer. DC measurements were performed utilizing a high-resolution source measurement unit (HRSMU), which was also employed to generate pulses for pulse testing. To transfer weights to the array, specific cells within the array were selected using a low-leakage switching matrix (Keysight E5250A) and a probe card. For inference, spikes were generated *via* the HRSMU, and at each time step, voltages of 1 V or 0 V were applied to 24 wordlines through the switching matrix, while the sum of currents was detected at bitlines set to ground. The overall processes of weight transfer and inference, including ISPVA, were controlled using a customized C++ program.

4.2. Pre-training simulations and lightweight networks

We first derived FP32-formatted weights trained using a software-based ANN. Before training, all weights of the network with dimensions of 784 × 24 × 10 were initialized using the He initialization method.[32] During training, we used the back-propagation algorithm, the cross-entropy loss function, and the stochastic gradient descent (SGD) optimizer. The trained weights were then quantized to 3-bits to generate quantized weights, and their accuracy was evaluated using 10 000 test images. In the case of quantization with pruning, we initially performed pruning at a rate of 40% before 3-bit quantization. Pruning was conducted in an unstructured method, and the final weights were also assessed for accuracy using 10 000 test images. To implement both excitatory and inhibitory synapses, we utilized differential pairs of weights by calculating G6–G1 to realize $G^{+}5$. For SNN inference, we employed rate coding to convert pixel values into spike counts for the network's input spikes. Accordingly, at each time step, the normalized pixel values of the images (ranging from 0 to 1) were input into ReLU based neurons, resulting in higher pixel values producing spikes of identical amplitude but with increased firing frequency. The input spikes and weights transformed through these processes for the second layer were utilized in RRAM crossbar arrays and PySpice simulations.

4.3. PySpice simulations

PySpice (version 1.5) was utilized in a Jupyter Notebook environment based on Anaconda. A 24 × 20 array was modeled

to implement differential pairs similar to a physical array, accounting for non-ideal phenomena by embedding parasitic elements such as line resistance and contact resistance. Pre-trained weights were converted into piecewise linear (PWL) functions using Python code to enable weight transfer simulations at each time step. Input spikes for inference were similarly converted to PWL signals using Python code, with input spikes of 1 V or 0 V applied to the wordlines at each time step. These input spikes were multiplied by the conductance stored in the array and summed as the output currents at bitlines maintained at 0 V. Furthermore, the error in each modeled device was determined based on the measurement results shown in Fig. 4b. Within the range of −4 nS to 4 nS, 94% of the distribution lies between −2 nS and 2 nS, and an independent normal distribution with a standard deviation of 1.06 nS was assigned to represent the device error. In the array model, each time a specific cell transfers its weight, a random value sampled from this distribution is added. Consequently, when the current is measured at the end of the bitline, the 24 individual errors combine, causing the total bitline error to follow a new normal distribution according to the central limit theorem. More details are shown in Fig. S6 (ESI†). By implementing not only the weight-transfer process but also the errors arising during inference in this way, we have significantly enhanced the reliability of the array model simulation.

Author contributions

H. Kim: conceptualization, investigation, software simulation, experiments, writing – original draft. Kyungho Hong: conceptualization, device and array fabrication. Sungjoon Kim: Experiments. Woo Young Choi: supervision, review & editing. Min-Hwi Kim: supervision, funding acquisition, review & editing.

Conflicts of interest

There are no conflicts to declare.

Data availability

The data supporting this article have been included as part of the ESI.†

Acknowledgements

This research was supported by the National R&D Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (RS-2023-00217539), and by the Mid-Career Researcher Program (NRF-2021R1A2C1007931).

Notes and references

1 O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, A. M. Umar, O. U. Linus, H. Arshad, A. A. Kazaure, U. Gana and M. U. Kiru, *IEEE Access*, 2019, **7**, 158820.
2 Z. Huang, S. Sun, J. Zhao and L. Mao, *Inf. Fusion*, 2023, **98**, 101834.
3 S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu and E. Chen, *Natl. Sci. Rev.*, 2024, **11**, nwae403.
4 J. Yang, H. Jin, R. Tang, X. Han, Q. Feng, H. Jiang, S. Zhong, B. Yin and X. Hu, *ACM Trans. Knowl. Discov. Data*, 2024, **18**, 32.
5 Q. Xia and J. J. Yang, *Nat. Mater.*, 2019, **18**, 309–323.
6 B. Chen, F. Cai, J. Zhou, W. Ma, P. Sheridan and W. D. Lu, 2015 IEEE International Electron Devices Meeting (IEDM), 2015, pp. 17.5.1–17.5.4.
7 H. Li, S. Wang, X. Zhang, W. Wang, R. Yang, Z. Sun, W. Feng, P. Lin, Z. Wang, L. Sun and Y. Yao, *Adv. Intell. Syst.*, 2021, **3**, 2100017.
8 X. Duan, Z. Cao, K. Gao, W. Yan, S. Sun, G. Zhou, Z. Wu, F. Ren and B. Sun, *Adv. Mater.*, 2024, **36**, 2310704.
9 A. Taherkhani, A. Belatreche, Y. Li, G. Cosma, L. P. Maguire and T. McGinnity, *Neural Networks*, 2020, **122**, 253–272.
10 H. Kim, S. Hwang, J. Park, S. Yun, J.-H. Lee and B.-G. Park, *IEEE Electron Device Lett.*, 2018, **39**, 630–633.
11 J. Fowers, K. Ovtcharov, M. Papamichael, T. Massengill, M. Liu, D. Lo, S. Alkalay, M. Haselman, L. Adams, M. Ghandi, S. Heil, P. Patel, A. Sapek, G. Weisz, L. Woods, S. Lanka, S. K. Reinhardt, A. M. Caulfield, E. S. Chung and D. Burger, A configurable cloud-scale DNN processor for real-time AI, 2018, pp. 1–14.
12 X. Zhang, V. Mohan and A. Basu, *IEEE Trans. Circuits Syst. II Express Briefs*, 2020, **67**, 816–820.
13 B. Max, M. Hoffmann, H. Mulaosmanovic, S. Slesazeck and T. Mikolajick, *ACS Appl. Electron. Mater.*, 2020, **2**, 4023–4033.
14 K. L. Wang, J. G. Alzate and P. K. Amiri, *J. Phys. D: Appl. Phys.*, 2013, **46**, 074003.
15 T. Mikolajick, M. H. Park, L. Begon-Lours and S. Slesazeck, *Adv. Mater.*, 2023, **35**, 2206042.
16 F. Aguirre, A. Sebastian, M. Le Gallo, W. Song, T. Wang, J. J. Yang, W. Lu, M.-F. Chang, D. Ielmini and Y. Yang, *et al.*, *Nat. Commun.*, 2024, **15**, 1974.
17 K. Moon, S. Lim, J. Park, C. Sung, S. Oh, J. Woo, J. Lee and H. Hwang, *Faraday Discuss.*, 2019, **213**, 421–451.
18 S. Yu and P.-Y. Chen, *IEEE Solid-State Circuits Mag.*, 2016, **8**, 43–56.
19 B. Yan, B. Li, X. Qiao, C.-X. Xue, M.-F. Chang, Y. Chen and H. H. Li, *Adv. Intell. Syst.*, 2019, **1**, 1900068.
20 J.-H. Ryu and S. Kim, *Chaos, Solitons Fractals*, 2020, **140**, 110236.
21 K. Moon, A. Fumarola, S. Sidler, J. Jang, P. Narayanan, R. M. Shelby, G. W. Burr and H. Hwang, *IEEE J. Electron Devices Soc.*, 2018, **6**, 146–155.
22 S. Kunwar, Z. Jernigan, Z. Hughes, C. Somodi, M. D. Saccone, F. Caravelli, P. Roy, D. Zhang, H. Wang, Q. Jia, J. L. MacManus-Driscoll, G. Kenyon, A. Sornborger, W. Nie and A. Chen, *Adv. Intell. Syst.*, 2023, **5**, 2300035.
23 J. A. Dean, *Lange's Handbook of Chemistry*, 1999.
24 S. Kim, K. Park, K. Hong, T.-H. Kim, J. Park, S. Youn, H. Kim and W. Y. Choi, *Adv. Mater. Technol.*, 2024, **9**, 2400063.
25 Y. Li, S. Long, Y. Liu, C. Hu, J. Teng, Q. Liu, H. Lv, J. Suñé and M. Liu, *Nanoscale Res. Lett.*, 2015, **10**, 1–30.

26 Y. Shi, L. Nguyen, S. Oh, X. Liu, F. Koushan, J. R. Jameson and D. Kuzum, *Nat. Commun.*, 2018, **9**, 5312.
27 H. Wu, P. Judd, X. Zhang, M. Isaev and P. Micikevicius, *arXiv*, 2020, preprint, arXiv:2004.09602, DOI: 10.48550/arXiv.2004.09602.
28 Y. Liao, B. Gao, F. Xu, P. Yao, J. Chen, W. Zhang, J. Tang, H. Wu and H. Qian, *IEEE Trans. Electron Devices*, 2020, **67**, 1593–1599.
29 M.-H. Kim, S.-H. Lee, S. Kim and B.-G. Park, *IEEE Access*, 2022, **10**, 37030.
30 M. Hu, C. E. Graves, C. Li, Y. Li, N. Ge, E. Montgomery, N. Davila, H. Jiang, R. S. Williams, J. J. Yang, Q. Xia and J. P. Strachan, *Adv. Mater.*, 2018, **30**, 1705914.
31 H. Zhao, Z. Liu, J. Tang, B. Gao, Y. Zhang, H. Qian and H. Wu, *Tsinghua Sci. Technol.*, 2022, **27**, 455–471.
32 K. He, X. Zhang, S. Ren and J. Sun, Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.