Scaling-up Resistive Synaptic Arrays for Neuro-inspired Architecture: Challenges and Prospect

Shimeng Yu[1*], Pai-Yu Chen[1], Yu Cao[1], Lixue Xia[2], Yu Wang[2], Huaqiang Wu[2]
[1]Arizona State University, Tempe, AZ 85281, USA, [2]Tsinghua University, Beijing, 100084, China,
[*]Email: shimengy@asu.edu

Abstract- The crossbar array architecture with resistive synaptic devices is attractive for on-chip implementation of weighted sum and weight update in the neuro-inspired learning algorithms. This paper discusses the design challenges on scaling up the array size due to non-ideal device properties and array parasitics. Circuit-level mitigation strategies have been proposed to minimize the learning accuracy loss in a large array. This paper also discusses the peripheral circuits design considerations for the neuro-inspired architecture. Finally, a circuit-level macro simulator is developed to explore the design trade-offs and evaluate the overhead of the proposed mitigation strategies as well as project the scaling trend of the neuro-inspired architecture.

Index Terms- Resistive memory, synaptic device, crossbar array, machine learning, neuromorphic computing

1. Introduction

Recent advances in neuro-inspired machine learning algorithms have shown tremendous success in the tasks such as image recognition when they are run on supercomputers [1]. However, sequential von Neumann architecture is inadequate for achieving real-time learning with a large dataset and under a low-power constraint. Although custom designed ASIC (e.g. IBM's TrueNorth [2]) with distributed memories has shown significant advantages over CPU/GPU, the SRAM based synaptic arrays still require sequential row-by-row read operation. To further parallelize the key operations in the learning algorithms such as weighted sum (or matrix-vector multiplication) and weight update, the crossbar array architecture with resistive synaptic devices has been proposed [3] (Fig. 1). Although the synaptic multilevel behavior has been reported in various resistive memories (one example shown in Fig. 2) at device level [4-6] and a simple learning algorithm has been experimentally demonstrated at small network level [7], extending the array size to a large-scale is a nontrivial task. In this paper, we discuss the design challenges of scaling up of the array size and present the potential solutions for overcoming these challenges.

2. Non-ideal Device Properties and Array Parasitics

When scaling up the array size, the non-ideal device properties and array parasitics may potentially degrade the learning accuracy [8]. The non-ideal properties of the realistic devices today include a finite weight precision, a nonlinearity in weight update (conductance vs. #pulse), limited on/off ratio and device variations, see device examples (Fig. 3-5). We build a device behavioral model of the weight update considering the extent of nonlinearity (Fig. 6) and device variations including both spatial (device-to-device) variations and temporal (cycle-to-cycle) variations (Fig. 7). We also build a SPICE simulator to study the IR drop problem and RC latency due to the interconnect resistance and parasitic capacitance in the array (Fig. 8). In order to quantify the impact on learning accuracy, we use the sparse coding [9] algorithm as a case study. Sparse coding is an unsupervised learning algorithm to extract the inherent feature vector (Z) from the dataset (X) through a dictionary matrix (D). We incorporate the device behavioral model into the weight update ($\Delta D \sim rZ$) and incorporate the array-level SPICE results into the matrix-vector multiplication (DZ) in the algorithm (Fig. 9). We use MNIST handwritten digits dataset [10] for the training, and the recognition accuracy is the metric for the following discussions: 1) to avoid a significant accuracy loss, a 6-bit D (64 multi-levels for resistive synaptic devices) is needed (Fig. 10), which is achievable in today's devices [4-6]. 2) the nonlinear weight update slightly decreases the accuracy by a few percentages (Fig. 11). 3) a small on/off ratio greatly degrades the learning accuracy (Fig. 12), thus some of today's devices [4-6] become problematic. 4) the algorithm can tolerate the spatial variations but has poor resilience against the temporal variations (Fig. 13). The nanoscale interconnect in the large array causes significant IR drop that distorts the matrix-vector multiplication thereby degrading the accuracy (Fig. 14). Potential circuit-level solutions to address the above issues include using a dummy column to eliminate the off-state current by differential read-out (Fig. 15), using multiple-cell (e.g. 3×3 cells) as one bit to average out the device variations (Fig. 16), and relaxing the wire width for reducing the IR drop along interconnect. With these strategies, the recognition accuracy can be brought back to 95% as compared to 65% in a naïve implementation (Fig. 17). The hardware overhead due to these strategies is evaluated in Section 4.

3. Peripheral Circuits Design Considerations

Different from the 1-transistor-1-resistor (1T1R) conventional

memory array (Fig. 18), we propose rotating BL to make it perpendicular to SL (Fig. 19). When all WLs are turned on, the transistors enter the deep triode region and become transparent, thus the rotated 1T1R array becomes a pseudo-crossbar. More aggressively, we can get rid of the transistors and have a true crossbar in the core (Fig. 20), however the cells may face the half-select problem during the programming. It is worth pointing out that the sneak path problem does not exists here because all the cells participate in the matrix-vector multiplication. Nevertheless the IR drop problem still exists. Another difference in the peripheral circuits design is that the decoder needs to enable multiple rows or columns, thus a switch matrix is used instead. One key component of the crossbar architecture is the read circuit that converts the analog column current to the digital output. We propose a read circuit [11] that employs the principle of the integrate-and-fire neuron model (Fig. 21-22). We have sent out the pseudo-crossbar design (64×64) with read circuits for tape-out (Fig. 23). It is designed in 130 nm CMOS with post-processing of resistive devices between M4 and M5 at the Tsinghua Univ. Fab.

4. Chip-level Design Explorations and Trade-offs

To facilitate the crossbar design space exploration, we develop a circuit-level macro simulator to estimate the area, latency, power given an array size and peripheral circuit technology node, following the principle of CACTI [12] for SRAM cache and NVSim [13] for non-volatile memory. The following analysis is based on 65 nm peripheral circuit technology node for a true crossbar array. In a small array (32×32), the peripheral circuits occupy a significant part of the area, while in a large array (1024×1024), the crossbar core dominates the area (Fig. 24). It should be noted that the large array becomes even larger because a more relaxed wire width is required to relieve the IR drop problem as discussed in Section 2 (Fig. 25). Considering the multiple-cell (3×3 cells) as one bit scheme for reducing the device variation, a large array (1024×1024) increases the area almost by 9×, while a small array (32×32) increases the area by 2× (Fig. 26). As the read circuit is a major area contributor, in a small array, multiple columns may share one read circuit to improve the area efficiency (Fig. 27), however, this may increase the read latency as multiple-read (or time multiplexing) is needed because of the sharing. In a large array, since the crossbar core dominates the area and part of the peripheral circuits can be even hidden underneath the crossbar core, one column can have its own read circuit, thus the fully parallel read becomes possible. However, the RC delay in a large array becomes noticeable, e.g. in a 1024×1024 array even with a relaxed wire width 100 nm (Fig. 28), the RC delay approaches 500 ps. Moreover, the write energy increases rapidly with the scaling up of the array size (Fig. 29) as a row-by-row write operation is assumed and the half-selected cells during the programming are contributing to the write energy. Finally, we project a scaling trend of the peripheral circuit technology node from 65 nm down to 14 nm (Fig. 30). The large array actually could not scale well because the area is dominated by the crossbar core, and the wire width of the core has to be relaxed for a large array due to the IR drop problem.

5. Conclusion

The design challenges on scaling up the crossbar array size for neuro-inspired architecture are caused by the non-ideal device characteristics and array parasitics (e.g. the IR drop problem). Circuit-level mitigation strategies have been proposed with affordable overhead of area, latency, and power. In the future work, an architecture-level mapping tool is to be developed to efficiently partition the array size given learning algorithms and dataset under the constraint of hardware resources.

Acknowledgement

This work is in part supported by NSF-CCF-1449653.

Reference

[1] Q. V. Le, et al. “Building high-level features using large scale unsupervised learning,” ICML, 2012.

[2] P. A. Merolla, et al. “A million spiking-neuron integrated circuit with a scalable communication network and interface,” Science, 2014.

[3] P.-Y. Chen, et al. “Technology-design co-optimization of resistive cross-point array for accelerating learning algorithms on chip,” DATE, 2015.

[4] I-T. Wang, et al. “3D synaptic architecture with ultralow sub-10 fJ energy per spike for neuromorphic computation,” IEDM, 2014.

[5] S. Park, et al. “Neuromorphic speech systems using advanced ReRAM-based synapse,” IEDM, 2013.

[6] S. H. Jo, et al. “Nanoscale memristor device as synapse in neuromorphic systems,” Nano Lett, 2010.

[7] M. Prezioso, et al. “Training and operation of an integrated neuromorphic network based on metal-oxide memristors,” Nature, 2015.

[8] P.-Y. Chen, et al. “Mitigating effects of non-ideal synaptic device characteristics for on-chip learning,” ICCAD, 2015.

[9] H. Lee, et al. “Efficient sparse coding algorithms,” NIPS, 2006.

[10] MNIST, http://yann.lecun.com/exdb/mnist/

[11] D. Kadetotad, et al. “Parallel architecture with resistive crosspoint array for dictionary learning acceleration,” JETCAS, 2015.

[12] CACTI, http://www.hpl.hp.com/research/cacti/

[13] NVSim, http://nvsim.org/

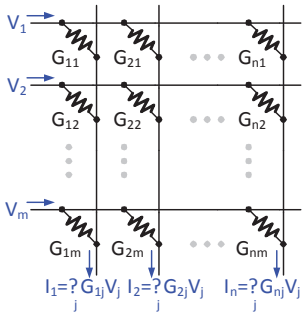

Fig. 1 The resistive crossbar array architecture for matrix-vector multiplication. Column output current sums up row input voltage weighted by the device conductance at each cross-point.

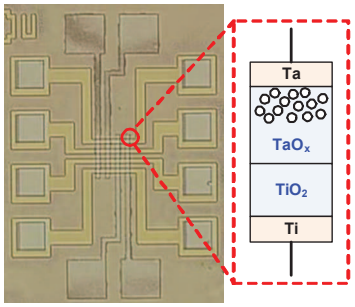

Fig. 2 Example of a small scale of crossbar array with TaO_x/TiO_2 resistive synaptic devices [4]. The oxygen vacancies migrate between the TaO_x and TiO_2 interface under voltage programming pulse thereby modulating the device conductance.

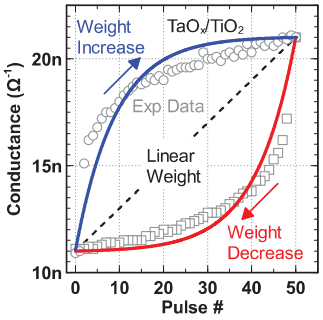

Fig. 3 The weight update curve for TaO_x/TiO_2 resistive synaptic devices [4]. Non-ideal properties include nonlinear weight update, limited on/off ratio, and variations.

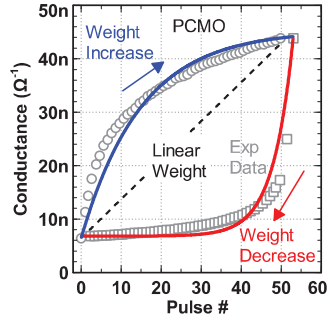

Fig. 4 The weight update curve for PCMO resistive synaptic devices [5]. Non-ideal properties include nonlinear weight update, limited on/off ratio, and variations.

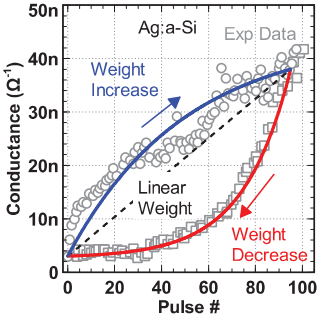

Fig. 5 The weight update curve for Ag:a-Si resistive synaptic devices [6]. Non-ideal properties include nonlinear weight update, limited on/off ratio, and variations.

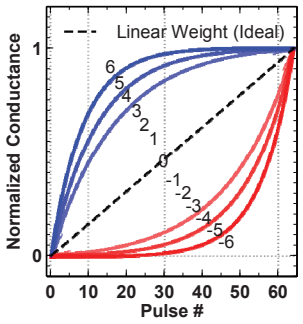

Fig. 6 The device behavioral model of the nonlinear weight update. The extent of nonlinearity is labeled from 1 to 6.

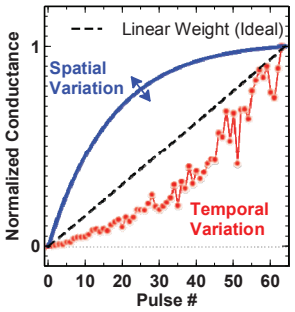

Fig. 7 The spatial (device-to-device) variation is modeled as a variation of the weight update baselines. The temporal (cycle-to-cycle) variation is modeled as a random noise.

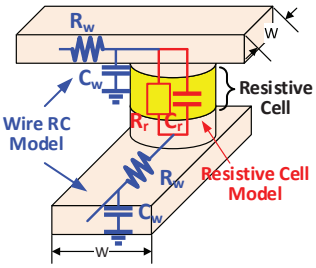

Fig. 8 The SPICE sub-circuit module for one resistive synaptic device at a cross-point in the array. The wire resistance and parasitic capacitance are included. The module is duplicated in 2D space for simulating the entire array.

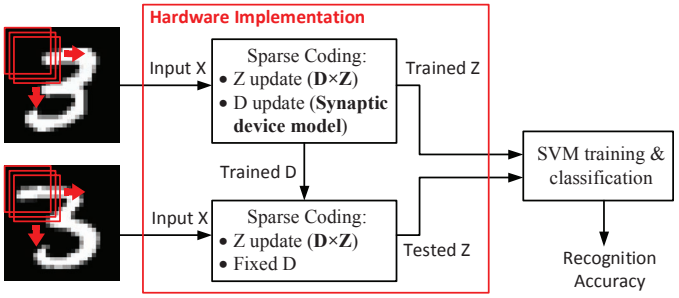

Fig. 9 The flow of training and testing using MNIST handwritten digits dataset [10]. The unsupervised sparse coding algorithm is used to extract the inherent features for the next-stage classification using support-vector-machine (SVM). The non-ideal device properties are included in the D update. The array parasitics calculated by SPICE are included in the DZ step in the Z update.

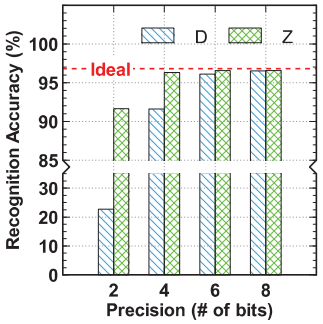

Fig. 10 The recognition accuracy as a function of the data precision of the D, Z values in the code. 6-bit D is required indicating 64-level is needed in the resistive devices.

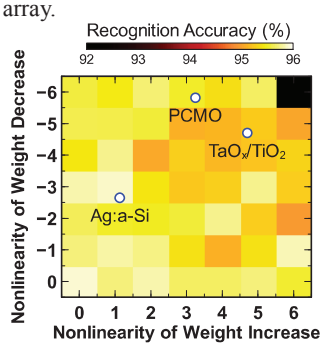

Fig. 11 The recognition accuracy loss due to the weight update nonlinearity. The realistic device data [4-6] is labeled, leading to a few percent drop of the accuracy.

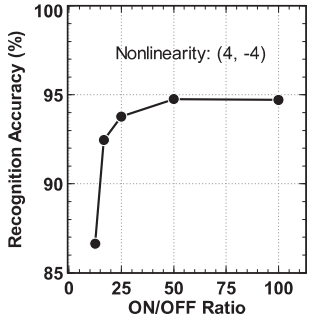

Fig. 12 The recognition accuracy loss due to the limited on/off ratio. A small on/off ratio introduces significant degradation. The realistic devices [4-6] may become problematic.

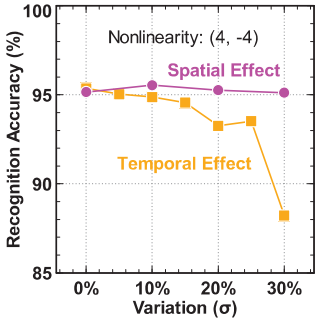

Fig. 13 The recognition accuracy loss due to the device variation. The algorithm can tolerate the spatial variation while it has poor resilience against the temporal variation.

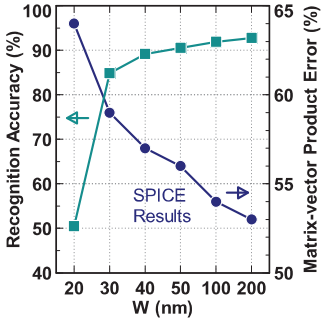

Fig. 14 The recognition accuracy loss due to the IR drop along the interconnect of the array. A relaxation of wire width is proposed to minimize the degradation.

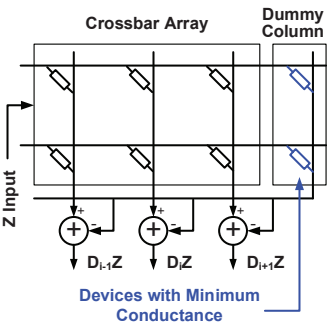

Fig. 15 The dummy column with minimum conductance cells is proposed to eliminate the off-state current and improve on/off ratio by a differential read-out.

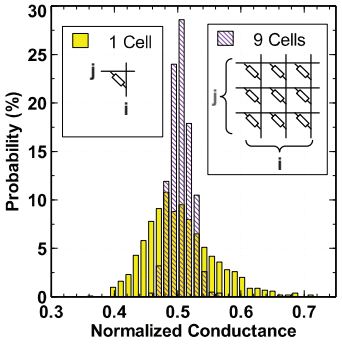

Fig. 16 Multiple-cell as one bit is proposed to statistically average out the device variations.

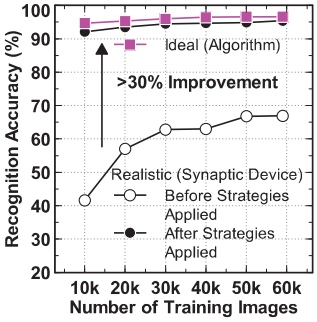

Fig. 17 The recognition accuracy improvement with the proposed strategies (relaxing wire, dummy column, multiple-cell as a bit).

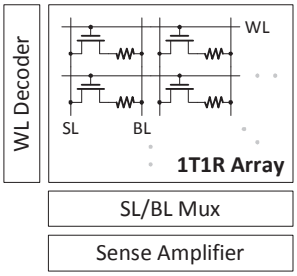

Fig. 18 The circuit block diagram for the conventional 1T1R memory array. BL and SL are in parallel. S/A is used.

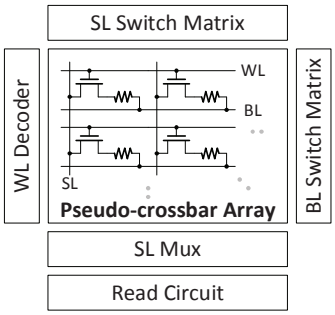

Fig. 19 The circuit block diagram for the pseudo-crossbar by rotating the BL in the 1T1R memory array. Switch matrix is used to enable multiple rows.

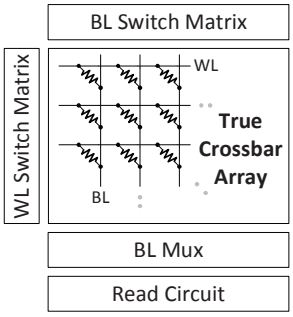

Fig. 20 The circuit block diagram for the true crossbar by removing the transistors in the array core. Special read circuit is needed to convert analog current to digital output.

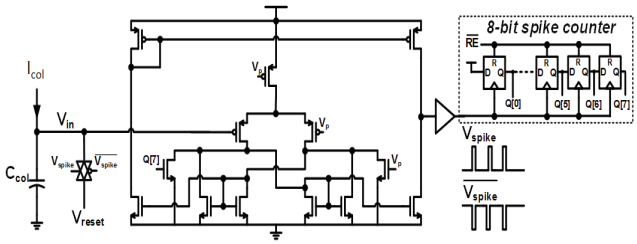

Fig. 21 Design of a read circuit [11] that employs the principle of the integrate-and-fire neuron model. Starting from a reset voltage, the column current is integrated on the finite capacitance of each column, when the voltage charges up above a certain threshold voltage, the output switches and the capacitance is discharged back to the reset voltage. The output of the Schmitt trigger is buffered and drives the clock input of a shift register to store the digital output data.

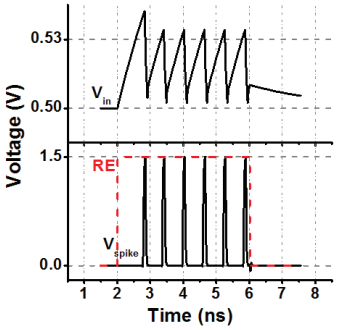

Fig. 22 The simulated waveform of the integrated input voltage and the digital output spike of the read circuit in 65 nm CMOS.

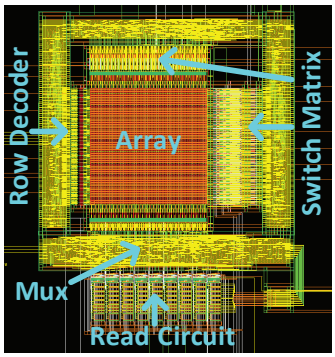

Fig. 23 The layout of the designed 64×64 pseudo-crossbar array with read circuits. The CMOS is in 130 nm and the resistive devices are fabricated between M4 and M5 by post-processing.

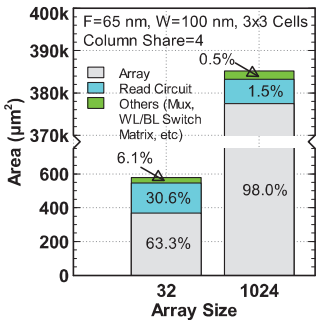

Fig. 24 The area breakdown of the crossbar array. In a small array (32×32), peripheral circuits occupy noticeable area. In a large array (1024×1024), the crossbar core dominates.

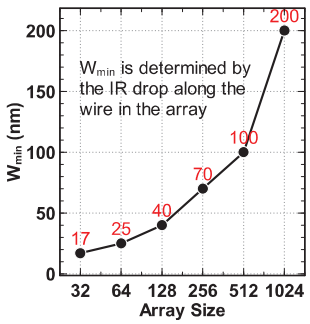

Fig. 25 The requirement of the minimum wire width for different array sizes. The minimum wire width is determined by the IR drop along the interconnect.

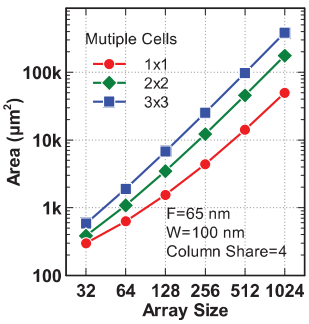

Fig. 26 The area overhead of the multiple-cell as one bit for different array sizes. The large array has proportional area overhead because the crossbar core dominates.

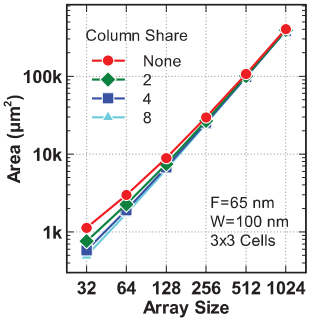

Fig. 27 The area overhead of column sharing for different array sizes. A large array does not need column sharing and can have one read circuit for one column, thus a fully parallel read is possible.

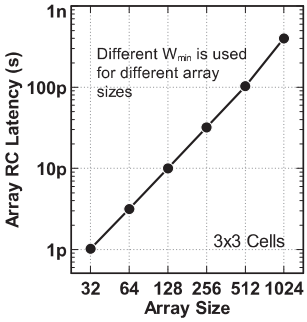

Fig. 28 The array RC delay due to the wire resistance and parasitic capacitance for different array sizes. A large array even with a relaxed wire width can have ~500 ps RC delay.

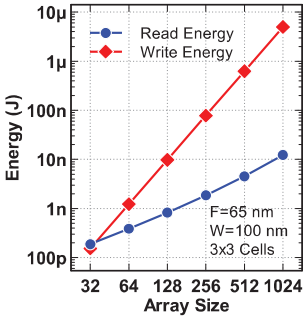

Fig. 29 The read/write energy consumption for different array sizes. A large array consumes much more write energy due to multiple row-by-row accesses in the write operation.

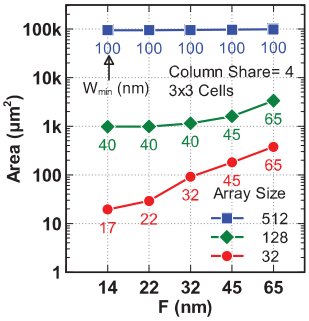

Fig. 30 The scaling trend of the crossbar array from peripheral circuit technology node 65 nm to 14 nm. The large array does not scale well because a relaxed wire width limits the array area despite of the peripheral circuits scaling.