Solid-State Electronics 227 (2025) 109128

Contents lists available at ScienceDirect

Solid State Electronics

journal homepage: www.elsevier.com/locate/sse

Performance of Pulse-Programmed memristive crossbar array with bimodally distributed stochastic synaptic weights☆

Nadine Dersch [a,b,*], Eduardo Perez [c,d], Christian Wenger [c,d], Christian Roemer [a,b], Mike Schwarz [a], Benjamin Iniguez [b], Alexander Kloes [a]

[a] NanoP, THM University of Applied Sciences, Giessen, Germany
[b] DEEEA, Universitat Rovira i Virgili, Tarragona, Spain
[c] IHP-Leibniz-Institut für innovative Mikroelektronik, Frankfurt (Oder), Germany
[d] BTU Cottbus-Senftenberg, Cottbus, Germany

ARTICLE INFO

The review of this paper was arranged by Francisco F. Gamiz

Keywords:
Artificial neural networks
Bimodal distribution
Memristive crossbar array
Noise-based simulation
Pulse-programming
Variability

ABSTRACT

In this paper, we present a method of implementing memristive crossbar arrays with bimodally distributed weights. The bimodal distribution is a result of pulse-based programming. The memristive devices are used for implementing synaptic weights and can only have an ON (logical "1″) or an OFF (logical ″0″) state. The state of the memristive device after programming is determined by the bimodal distribution. The highly efficient noise-based variability approach is used to simulate this stochasticity. The memristive crossbar array is used to classify the MNIST data set and comprises more than 15,000 weights. The interpretation of these weights is investigated. In addition, the influence of the stochasticity of the weights and the accuracy of the weights on the classification results is considered and various programming settings are examined.

1. Introduction

Memristive devices (MDs) are non-volatile memories and are considered promising candidates for the development of hardware-based artificial neural networks (ANNs) [1,2]. The MDs can be in one of the two states: low-resistive-state (LRS) and high-resistive-state (HRS) [3]. During the SET process, the MD is switched to LRS, which corresponds to a logical "1″. The RESET process switches the MD to HRS, which corresponds to a logical ″0″. Beyond this basic binary switching behavior [4] MDs can be programmed in multiple intermediate conductance levels (multi-level operation) [5,6]. Only the binary approach is considered in this work. The MDs exhibit stochastic fluctuations which result in device-to-device and cycle-to-cycle variability [7]. This stochastic variability can be simulated using the Noise Based Variability Approach (NOVA) [5]. Synaptic weights of ANNs can be implemented as a memristive crossbar array, whereby a single cell, consisting of two MDs, functions as a weight with possible values from -1 to $+1$ [8].

2. Setup of the memristive crossbar array and programming Scheme

To classify the MNIST data set (images consisting of 28x28 pixels), the ANN consists of 784 inputs, 10 outputs and thus 15,680 memristive cells (as in [5]). In the simulation, the MDs are considered as simple fluctuating resistors for simplicity, whereby their conductance and its variability has been extracted from measurements on oxide-based MDs [4]. Two memristive cells with certain conductances G are required to design a weight W between -1 and $+1$: one G^+ and one G^- device [5,6], whereby the weight is given by

$$W = G^+ - G^- \tag{1}$$

The input data is applied positively to G^+ and negatively to G^- in the form of voltage values (pixel values are converted). The current result in each MD depends on its conductance G^+ or G^-. These currents are then added together for classification. This results in a total current for each output. The required weight values come from the software training. The memristive cells are programmed by applying pulses, which can be

☆ This article is part of a special issue entitled: 'EuroSOI-ULIS 2024' published in Solid State Electronics.

* Corresponding author.
E-mail addresses: nadine.dersch@ei.thm.de (N. Dersch), alexander.kloes@ei.thm.de (A. Kloes).

https://doi.org/10.1016/j.sse.2025.109128

Available online 24 April 2025

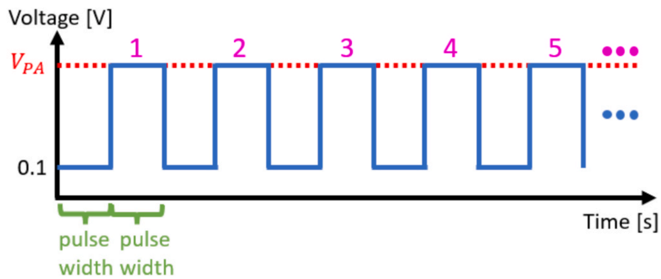

Fig. 1. Depiction of the programming pulses, which can be varied in amplitude (red), pulse width (green) and number (pink).

changed in terms of amplitude, pulse width and number of pulses [4]. This is shown in Fig. 1. The values of the weights result from their switching probability. The MDs can only become the logical values "0" and "1" and their state changes with a certain probability depending on the pulses applied. Accordingly, their conductivity follows a bimodal distribution (see Fig. 2 (a) and (c) in black). For each applied pulse, the state of the memristive cell can be represented via a bimodal distribution (probability for the device being in the HRS or in the LRS) [9]. In [10] it is shown that the statistical variation resulting from a superposition of many bimodal distribution functions can be represented by the superposition of Gaussian distribution functions. This allows the replacement of the bimodal distributions with Gaussian distributions for the usage of NOVA to simulate the fluctuations resulting from a large number of MDs in a crossbar array. The simulations are carried out with the Spectre simulator *Cadence Virtuoso* [11]. After the simulation, the winner is determined according to the winner-takes-all principle as in [12].

3. Interpretation of the weight Definition from pulse programming

A pulse with a pulse width of 1 µs and an amplitude of 0.8 V is defined for programming the devices (see Fig. 2 (a)). This pulse is sent max. 100 times to 128 different cells that are in the HRS before the first pulse. A MD can therefore be in the HRS state or in one of the 100 programming states depending on the number of applied pulses (measurement data from [4]). To be able to use NOVA, an average value and a standard deviation are calculated for each programming state using the 128 measured devices. According to equation (1), two MDs are required for a weight W, whereby a weight W can be composed from the average values of the bimodal distribution functions of G^+ and G^- (see Fig. 3 (a)).

Paths (e.g. C_1 and C_2) as in [6] can be formed within this matrix, which must be set for the range from -1 to $+1$ for W. At least G^+ or G^- have the lowest conductance for C_1 and the highest conductance for C_2. The reason for this choice is that when comparing these two paths, the highest possible contrast in respect to power consumption of the settings is considered. However, it is noticeable that the values of G^+/G^- do not increase with a constant step size, which means that the values from -1 to $+1$ can also be set with different accuracy. This is illustrated in Fig. 3 (b). Here the path C_1 covers weight values in the range from -1 to -0.5 and $+0.5$ to $+1$ with high accuracy, whereby C_2 provides a higher resolution in the range from -0.5 to $+0.5$.

Fig. 4 shows the standard deviation of each possible weight W. Here it can be seen that path C_1 has lowest standard deviations and C_2 has the highest standard deviations.

4. Simulation results of the memristive crossbar array

The memristive crossbar array is tested with the same images of a "7" and a "4" as from [5] (programming of weights by conductance level), as well as the image of a "1" (see Fig. 5). For all cases, the weights are trained with a resolution of 0.1 wt stepping. The paths C_1 and C_2 are compared by adjusting the target weight to the closest possible value (see Fig. 3 (b)). The results are shown in Table 1, as percentage of classifying the given number as a winner.

Table 1 shows that path C_2 delivers significantly worse classification results than path C_1. The reason for this is that the variability in the weights is very large for path C_2, which means that no precise classification is possible. The three digits are all classified with a similar probability. In contrast, in path C_1 the "7" is classified correctly with 87.84 % (in [5] in the worst case 99.4 % and best case 100 %) and the "1" with 76.74 %. The "4" is misclassified in most cases and is most frequently identified as a 7. In [5], this "4" is correctly classified in the best case with 50.44 %.

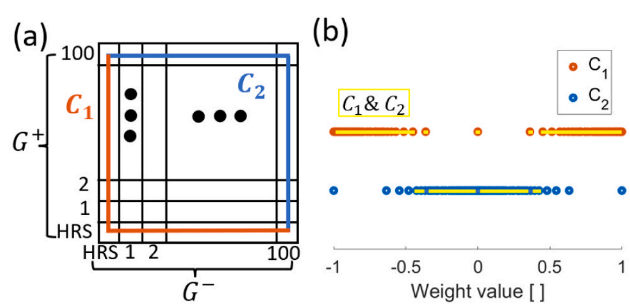

Fig. 3. (a) Representation of the 10,201 possible weights within the -1 to $+1$ range in the 101x101 matrix via paths. (b) Possible discrete weight values within the -1 to $+1$ range depending on the selected path.

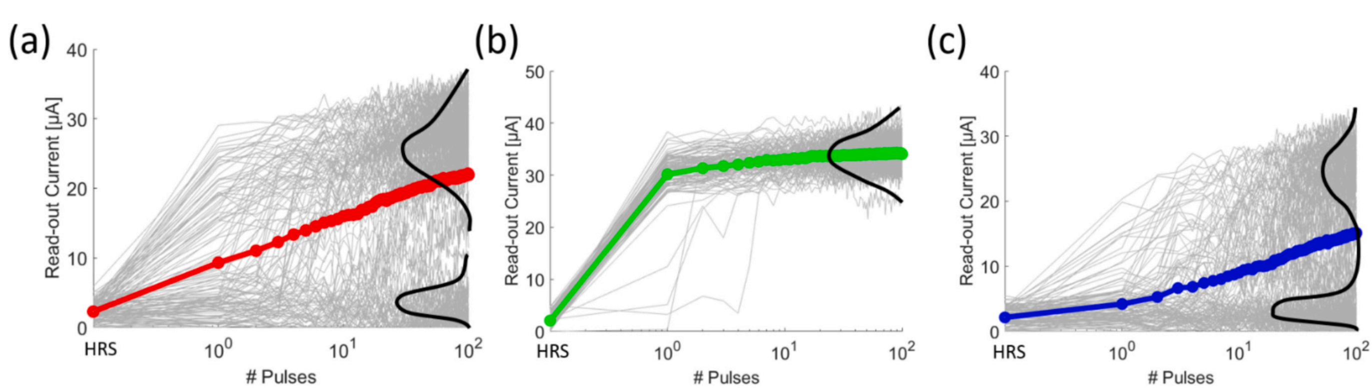

Fig. 2. (a) Programming with 100 pulses, 0.8 V amplitude and a pulse width of 1 µs, (b) Programming with 100 pulses, 1.1 V amplitude and a pulse width of 1 µs, (c) Programming with 100 pulses, 0.9 V amplitude and a pulse width of 100 ns. The distribution function for the last pulse is shown in black. The current values of 128 different cells are shown in gray and their mean values are colored.

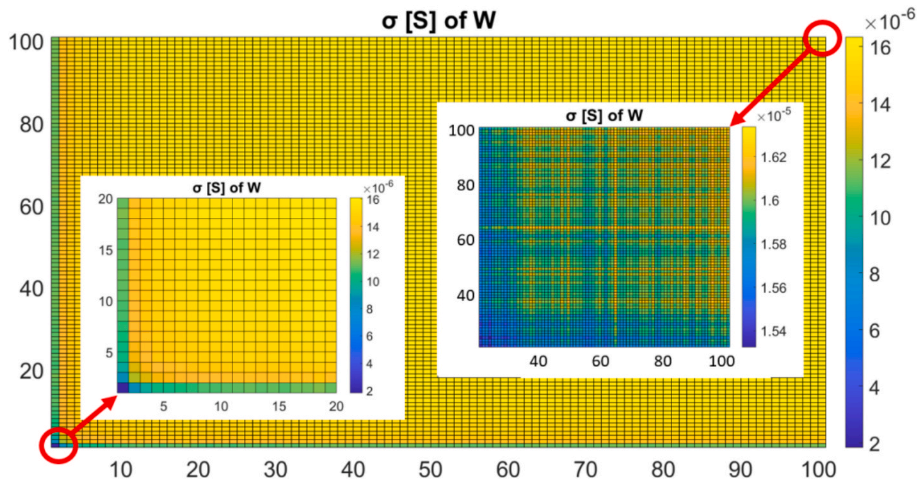

Fig. 4. Visualization of the resulting standard deviation for each possible weight to be set (101 states x 101 states). Zoom on the left shows even more clearly the standard deviation of the weights created with the help of the first 20 states. Zoom on the right shows with a new color scale how the standard deviation changes with the weights composed from the last 81 states.

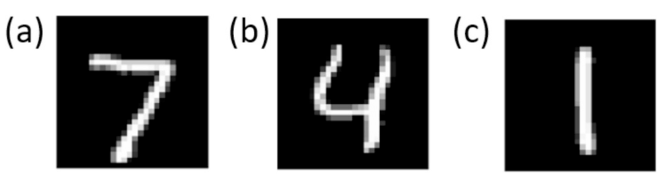

Fig. 5. Test images of the MNIST data set: (a) “7″, (b) “4” and (c) “1”.

5. Influence of the weight stepping and the variations

Two factors play a role in the result of the programming: 1) The possible fluctuation of the desired weight value and 2) the number of adjustable weights or the accuracy of how finely resolved they should be set.

Path C_1 was selected for testing the “1″ image, with weights set to increments of 0.1, 0.05 and 0.025 and noise levels of 100 % (referred to the statistical variations as observed in the measurements), 85 %, 67 %, 25 % and 0 %. The results are shown in Fig. 6.

Fig. 6 shows that the classification with a step of 0.025 is better than with 0.1, but the result only improves slightly. However, the 0.05 step is worse than 0.1, as the finer discretization may lead to weight combinations with an increased σ. It is noticeable that a reduction of the variability provides significantly better classification results. Even with a reduction of 1/3, the results are in the 90 % range for all step sizes.

A reduction in the stochasticity therefore has more influence on the classification result than the weight stepping. This should be checked by taking the weights of paths C_1 and C_2 from Fig. 3 (b) together so that there are no gaps in the weight setting. The simulation results can be found in Table 2. In this instance, the programming from Fig. 2 (a) was set, the test images with the “7″, “1” and “4” were examined and the weight stepping were set to 0.1.

Table 2 shows that paths C_1 and C_2 together (this setting is called C_1 & C_2 and ensures a weight range from −1 to + 1, as shown in yellow in Fig. 3 (b)) provide a better result than C_2, but worse than C_1 for “7″ and ”1″. This shows that setting less accurate weights from −1 to −0.5 and + 0.5 to + 1 (C_1 only) provides more correct expected results than setting more accurate weights from −1 to + 1 (C_1 & C_2). The cause is: Weights with high stochasticity as in C_2 lead to more incorrect results than weights with low stochasticity as in C_1. Even if only some of the weights have high stochasticity. This shows that the stochasticity has a major influence.

Table 1
Classification results of the images of a “7″, ”4″ and “1″ with paths C_1 and C_2. The expected result is marked in green and the result with the highest probability is marked in blue.

Results	"7", C_1	"4", C_1	"1", C_1	"7", C_2	"4", C_2	"1", C_2
0	4.65%	23.94%	2.81%	7.73%	9.36%	4.45%
1	1.35%	6.94%	76.74%	6.28%	8.64%	15.48%
2	0.44%	0.36%	0.08%	11.17%	10.79%	9.13%
3	2.57%	1.10%	4.34%	12.11%	7.67%	11.00%
4	1.13%	8.52%	4.60%	8.14%	11.69%	8.57%
5	0.24%	0.88%	0.21%	9.73%	9.62%	9.39%
6	0.84%	16.15%	4.35%	6.00%	10.30%	10.60%
7	87.84%	32.58%	6.23%	20.68%	11.20%	12.00%
8	0.92%	9.39%	0.34%	9.74%	12.30%	10.26%
9	0.02%	0.14%	0.30%	8.42%	8.43%	9.12%

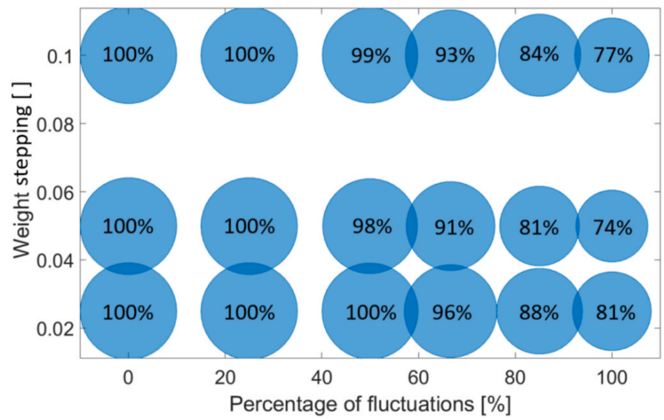

Fig. 6. Illustration of the effect of different increments of the weights and a reduction in the variability of the weights. The percentage for the correct classification of "1″ is given.

Table 2
Classification results of the images of a "7″, ″4″ and "1″ with paths C_1, C_2 and C_1 & C_2.

Setting	"7″	"4″	"1″
C_1	87.84 %	8.52 %	76.74 %
$C1_2$	20.68 %	11.69 %	15.48 %
C_1 & C_2	23.42 %	12.30 %	17.15 %

6. Different programming settings

Now three different pulse settings were tested to investigate the effect of different programming pulses. The amplitudes and pulse widths were:

1) 0.8 V and 1 μs (see Fig. 2 (a)): Programming as in the previous section. At the end of the sequence of pulses a MD is most likely ON.

2) 1.1 V and 1 μs (see Fig. 2 (b)): With just one pulse, the state of the MD is already very likely to change to ON.

3) 0.9 V and 100 ns (see Fig. 2 (c)): At the end of the sequence of pulses some MDs are already in the ON state and others remain in the OFF state.

The results of the simulation can be seen in Table 3 for path C_1 and C_2, each for the image of a "7″ and the weight stepping equal to 0.1.

Table 3 shows that the choice of programming conditions is important. With (a), the measuring points are similarly distributed on average and a good result is achieved. In contrast, with (b) only the ON states can be represented very accurately, which means that the classification result is significantly worse. With (c), large stochasticity can be observed and mainly the range of the OFF state can be set, which also makes classification difficult. As many measuring points in the range of both bimodal distributions are important, it is also important that there is some stochasticity (path C_1 with less stochasticity provides better results than path C_2 with more stochasticity). One solution could be to use different pulse widths, heights and numbers depending on the weight. In addition, it would be helpful if there were a greater distance between the HRS values and the LRS values expressed in current values. Another solution would be to extend the structure of the simple perceptron. The use of hidden layers could be helpful to counteract this.

7. Conclusions

The programming via pulses shows strong variability in the weight values. As a result, the classification accuracy of the MNIST dataset is influenced by the variability depending on the G^+/G^- settings. For these reasons, at least one of the MDs should always be in the HRS. This reduces the variations. In addition, for achieving a high probability for a correct classification, it is more important that the individual weights show less fluctuation than whether their conductivity can be set precisely. A reduction of the fluctuations observed in measurements by 33 % already shows significant improvements. However, even without fluctuations a precise setting of conductance states is important to achieve the correct classification results. The choice of programming is also important. Adjusting the pulse according to the required weight could be helpful to set the desired settings with the smallest possible fluctuations. The technology should also be designed in such a way that the HRS and LRS are clearly distinguishable in terms of the flowing current. In addition, the supplement of Hidden Layer will be useful to compensate for the fluctuations.

CRediT authorship contribution statement

Nadine Dersch: Writing – original draft, Visualization, Validation, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Eduardo Perez:** Writing – review & editing. **Christian Wenger:** Writing – review & editing. **Christian Roemer:** Writing – review & editing. **Mike Schwarz:** Writing – review &

Table 3
Classification results of the image of a "7″ with programming conditions (a), (b) and (c) for path C_1 and C_2. The expected result is marked in green and the result with the highest probability is marked in blue.

Results	(a) "7", C_1	(a) "7", C_2	(b) "7", C_1	(b) "7", C_2	(c) "7", C_1	(c) "7", C_2
0	4.65%	7.73%	10.00%	0.51%	2.14%	8.33%
1	1.35%	6.28%	10.00%	0.88%	2.32%	7.45%
2	0.44%	11.17%	10.00%	19.50%	19.13%	11.04%
3	2.57%	12.11%	10.00%	20.10%	14.92%	11.61%
4	1.13%	8.14%	10.00%	3.91%	4.03%	8.92%
5	0.24%	9.73%	10.00%	9.91%	7.72%	10.07%
6	0.84%	6.00%	10.00%	0.16%	0.65%	7.13%
7	**87.84%**	**20.68%**	10.00%	**28.66%**	**30.95%**	**16.19%**
8	0.92%	9.74%	10.00%	7.70%	9.88%	10.02%
9	0.02%	8.42%	10.00%	8.67%	8.26%	9.24%

editing. **Benjamin Iniguez:** Writing – review & editing, Supervision. **Alexander Kloes:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the Federal Ministry of Education and Research of Germany under the grant 16ME0092.

Data availability

Data will be made available on request.

References

[1] Zambelli C, Grossi A, Olivo P, Walczyk D, Dabrowski J, Tillack B, et al. *IEEE 6th International Memory Workshop (IMW)*. Taipei, Taiwan 2014;2014:1–4. https://doi.org/10.1109/IMW.2014.6849387.

[2] Huang P, Zhu D, Chen S, Zhou Z, Chen Z, Gao B, et al. Compact model of HfOX-based electronic synaptic devices for neuromorphic computing. *IEEE Trans Electron Devices* Feb. 2017;64(2):614–21. https://doi.org/10.1109/TED.2016.2643162.

[3] Jiang Z, Wu Y, Yu S, Yang L, Song K, Karim Z, et al. A compact model for metal–oxide resistive random access memory with experiment verification. *IEEE Trans Electron Devices* May 2016;63(5):1884–92. https://doi.org/10.1109/TED.2016.2545412.

[4] Perez E, Mahadevaiah MK, Perez-Bosch Quesada E, Wenger C. In-depth characterization of switching dynamics in amorphous HfO2 memristive arrays for the implementation of synaptic updating rules. Jpn J Appl Phys 2022;vol. 61, no. SM:1007. https://doi.org/10.35848/1347-4065/ac6a3b.

[5] N. Dersch, E. Perez-Bosch Quesada, E. Perez, C. Wenger, C. Roemer, M. Schwarz, et al., "Efficient circuit simulation of a memristive crossbar array with synaptic weight variability", Solid-State Electronics, vol. 209, p. 108760, ISSN 0038-1101, 2023, doi: 10.1016/j.sse.2023.108760.

[6] Milo V, Glukhov A, Pérez E, Zambelli C, Lepri N, Mahadevaiah MK, et al. Accurate program/verify schemes of resistive switching memory (RRAM) for in-memory neural network circuits. *IEEE Trans Electron Devices* Aug. 2021;68(8):3832–7. https://doi.org/10.1109/TED.2021.3089995.

[7] A. Kloes, C. Bischoff, J. Leise, E. Perez-Bosch Quesada, C. Wenger, and E. Perez, "Stochastic switching of memristors and consideration in circuit simulation," Solid-State Electronics, vol. 201, p. 108606, ISSN 0038-1101, 2023, doi: 10.1016/j.sse.2023.108606.

[8] Zambelli C, Grossi A, Olivo P, Walczyk D, Bertaud T, Tillack B, et al. Statistical analysis of resistive switching characteristics in ReRAM test arrays. In: in *2014 International Conference on Microelectronic Test Structures (ICMTS)*; 2014. p. 27–31. https://doi.org/10.1109/ICMTS.2014.6841463.

[9] Wenger C, Zahari F, Mahadevaiah MK, Pérez E, Beckers I, Kohlstedt H, et al. Inherent stochastic learning in CMOS-integrated HfO2 arrays for neuromorphic computing. *IEEE Electron Device Lett* April 2019;40(4):639–42. https://doi.org/10.1109/LED.2019.2900867.

[10] Dersch N, Roemer C, Perez E, Wenger C, Schwarz M, Iniguez B, et al. *IEEE Latin American Electron Devices Conference (LAEDC)*. Guatemala City, Guatemala 2024; 2024:1–4. https://doi.org/10.1109/LAEDC61552.2024.10555829.

[11] *Spectre (R) Circuit Simulator*, *Version 19.1.0.063*, Cadence Design Systems, Inc., 2019.

[12] N. Bogun, E. Perez-Bosch Quesada, E. Perez, C. Wenger, A. Kloes and M. Schwarz, "Analytical calculation of inference in memristor-based stochastic artificial neural networks," *2022 29th International Conference on Mixed Design of Integrated Circuits and System (MIXDES)*, Wrocław, Poland, 2022, pp. 83-88, doi: 10.23919/MIXDES55591.2022.9838321.