Reconfigurable Mapping Algorithm based Stuck-At-Fault Mitigation in Neuromorphic Computing Systems

Md. Oli-Uz-Zaman
University of South Alabama
Mobile, Alabama, USA
mo2025@jagmail.southalabama.edu

Saleh Ahmad Khan
University of South Alabama
Mobile, Alabama, USA
sk2021@jagmail.southalabama.edu

William Oswald
University of South Alabama
Mobile, Alabama, USA
wdo1621@jagmail.southalabama.edu

Zhiheng Liao
North Dakota State University
Fargo, North Dakota, USA
zhiheng.liao@ndsu.edu

Jinhui Wang
University of South Alabama
Mobile, Alabama, USA
jwang@southalabama.edu

ABSTRACT

Stuck-At-Fault (SAF) defect of memristor generated from immature fabrication and heavy device utilization makes neuromorphic computing systems commercially unavailable. To mitigate this problem, a Reconfigurable Mapping Algorithm (RMA) is proposed in this paper. Based on the analysis for the VGG8 model with CIFAR10 dataset, the experiment results show that the RMA is efficient in restoring the inference accuracy up to 90% (the original accuracy without SAF) under SAFs from 0.1% to 50%, where Stuck-At-One (SA1): Stuck-At-Zero (SA0) = 5:1, 1:5, and 1:1. Additionally, the RMA improves the accuracy more than 50% in presence of high nonlinearity LTP = 4 and LTD = -4 and the standard conductance drift (10 years at 85 degrees Celsius) nearly has no influence on the inference accuracy of the DNN with the RMA.

CCS CONCEPTS

• **Hardware → Analysis and design of emerging devices and systems**.

KEYWORDS

memristor, deep neural network (DNN), neuromporphic computing system, stuck-at-fault (SAF), inference accuracy

ACM Reference Format:
Md. Oli-Uz-Zaman, Saleh Ahmad Khan, William Oswald, Zhiheng Liao, and Jinhui Wang. 2023. Reconfigurable Mapping Algorithm based Stuck-At-Fault Mitigation in Neuromorphic Computing Systems. In *Proceedings of the Great Lakes Symposium on VLSI 2023 (GLSVLSI '23), June 5–7, 2023, Knoxville, TN, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3583781.3590208

1 INTRODUCTION

Nowadays, a DNN (Deep Neural Network) model deployed on neuromorphic computing systems is more popularThis is because DNNs have achieved a tremendous success due to their unparallel performance in important applications, such as computer vision, image processing, natural language processing, etc. [5] and meanwhile neuromorphic computing systems are highly effective in the Internet of Things (IoT) systems to perform computation, communication, and storage functions. However, with the increasing customer demand, DNN structures become more complex and require huge computational resources. Since the downscaling of the conventional CMOS technology is coming to the plateau, the CMOS-based neuromorphic computing devices are facing insurmountable challenges to deal with such DNN problems, because the computing speed of the CMOS-based neuromorphic computing systems has no space to improve and cannot further accelerate AI tasks. This situation creates an undesirable standstill towards the further advancement of the neuromorphic computing systems.

As the emerging non-volatile memory, memristors can be a rescuer from this deadlock. Besides non-volatility property, memristor exhibits wonderful characteristics like multilevel resistive state, low computational complexity [13], sub-nanosecond switching speed [16], sub-10-nm scalability [9], low energy dissipation of few pJ per bit [14, 16], long write-erase endurance [12] and CMOS-compatibility [7]. Additionally, memristor enables Compute-In-Memory (CIM) where the memory would be integrated into processing task to boost the system. The crossbar architecture and multilevel cell storage (multiple bits per cell) of memristors can very efficiently perform the vector matrix multiplication as the CIM, which is the most pivotal operation in the DNN algorithm.

Although memristor exhibits excellent properties, Stuck-At-Fault (SAF) in memristors causes reliability issue. SAF denotes a device when the resistance of a memristor freezes at High Resistive State (HRS) or Low Resistive State (LRS) [2]. Since the resistance variation is directly related to the mapped weights, the defective memristor will provide wrong weight and result in the inference error to the output of the DNN. To increase the immunity against SAF defect, several works have been proposed so far [3, 15, 18, 20]. These proposed hardware-based solutions have some limitations. Most of them use a complex algorithm to detect the defective memristors and most valuable weights first. Then a complex control circuit prevents those significant weights to be mapped to the defective cells. However, random patterns in SAF require individual optimization for each memristor array that is impossible when it comes to mass memristor-based computing device production.

GLSVLSI '23, June 5–7, 2023, Knoxville, TN, USA

ACM ISBN 979-8-4007-0125-2/23/06…$15.00
https://doi.org/10.1145/3583781.3590208

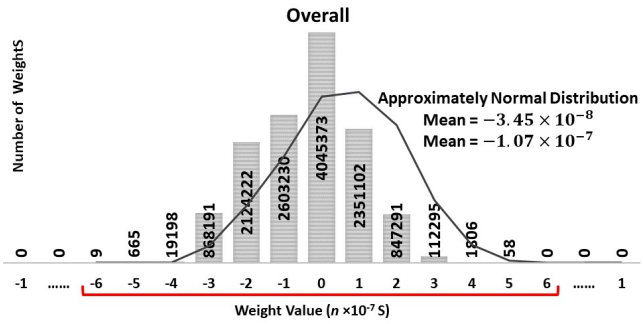

Figure 1: Overall Weight Distribution of VGG8 Model Trained by CIFAR10 Dataset

So, in this paper, a new technology – Reconfigurable Mapping Algorithm (RMA) is proposed in this paper to mitigate the influence of SAFs without existing limitations of previously proposed techniques.

2 METHODOLOGY

2.1 Stuck-At-Fault (SAF)

When the resistance state of the memristor is stuck at LRS, it is known as SA0 defect. A newly fabricated memristor possesses extremely high resistance. So, a forming process is required at the wafer level to initialize the memristor for regular read/write operations. The forming process is an action of inserting high tester voltage in every memristor for decreasing the resistance level to the normal LRS for around 100 us. The delicate insulator layer in the metal-insulator-metal (MIM) structure can severely be compromised during this process. Thus, some memristors would be overly formed because of variations in the unstable taster voltage and the thickness of the insulator layer. The resistance of the overly formed memristors stay at LRS forever and input stimulus pulse(s) fail(s) to change its resistance state.

On the other hand, when the resistance state of the memristor is stuck at HRS, it is known as SA1 defect. Word line (WL), bit line (BL), and select line (SL) are the three terminals to access each memristor inside the memristor crossbar architecture. But the broken WL makes memristor cells inaccessible for new write operation. Broken WL creates an open circuit where the resistance is unlimited. When the read circuit tries to read the resistance of those memristors, it always mistakes the cell as HRS.

It has been found that 9.04% and 1.75% memristor cells are affected by SA1 and SA0 respectively, which is approximately SA1:SA0=5:1 [2]. But this is not always the case. This ratio may vary from device to device. It has also been mentioned in [2] that over forming can cause 60% of the memristor cells to be SA0.

2.2 Weight Distribution

To validate the effectiveness of the memristor-based neuromorphic computing system, a VGG8 model along with CIFAR10 dataset are used. Fig. 1 shows the overall weight distribution of VGG8 model trained by CIFAR10 dataset. VGG8 model contains 12.97 million synapses to represent weights. Among the 12.97 million synaptic weights, 43.38% is negative, 25.53% is positive, and 31.18% is neutral weight. As listed in Fig. 1, the mean values of the layers are inclined towards zero and the standard deviations implies that those values are clustered closely. Besides, 99% of the weights are situated within $\pm(3 \times 10^{-7})$.

2.3 Reconfigurable Mapping Algorithm (RMA)

After analyzing the weight distribution in Fig. 1, an Reconfigurable Mapping Algorithm (RMA) is proposed in this paper. SAF causes huge discrepancy between the original weight and mapped weight. Hence, the inference accuracy degrades significantly even with a very small amount of SAFs inside the memristor crossbar array. The RMA maximally avoids the negative impact of SAF cells for systems and bring back the high accuracy.

RMA When SA1:SA0 = 5:1. According to conventional mapping, weights from the algorithm ranging from [-1, 1] will be mapped to memristor devices based on the resistance level [LRS, HRS]. The RMA will use the same resistance level [LRS, HRS], but it rearranges the same weights between [0, 1]. When SA1:SA0 = 5:1, most of SAFs are SA1. The algorithm for the RMA is as follows.

$$W_a = \begin{cases} 1 & W \geq 0, \\ 1 - |W| & W < 0, \end{cases} \quad (1) \qquad W_b = \begin{cases} 1 - W & W > 0, \\ 1 & W \leq 0, \end{cases} \quad (2)$$

where, W_a and W_b are the two positive portions of a single weight stored in two different memristors. The desired weight is extracted from the simple subtraction (W_a-W_b) during the execution. For example, when SA1:SA0 = 5:1, the RMA splits the original weight 0.3 into 1 and 0.7 according to equation 1, and 2. Those two weights will be mapped in two memristors and an op-amp based subtractor will extract the original weight (1-0.7) during the execution. This process creates huge number of "1". After mapped those "1", most of them replaces most of SA1s. In this case, the accuracy degradation can be greatly suppressed.

RMA When SA1:SA0 = 1:5. When SA1:SA0 = 1:5, SA0s are dominant in the memristor crossbar array. The algorithm for the RMA is as follows.

$$W_a = \begin{cases} W & W \geq 0, \\ 0 & W < 0, \end{cases} \quad (3) \qquad W_b = \begin{cases} 0 & W > 0, \\ |W| & W \leq 0, \end{cases} \quad (4)$$

Same as before, each weight is split into two positive weights W_a and W_b, and be mapped to two different memristors. During the execution, the subtractor subtracts the two weights (W_a-W_b) and bring back the original algorithmic weight. For example, negative weight -0.3 is split into 0 and +0.3 and is mapped in two memristors. At the end, the op-amp based subtractor generate (0-0.3), and the original weight (-0.3) is brought back. Since the conventional mapping takes place between [-1, 1] according to the resistance state [LRS, HRS], memristors stuck at LRS always report -1. But, the range of weight values are changed to [0, 1] with respect to the same resistance level [LRS, HRS] after applying the RMA. So the newly programmed memristors will provide "0" when it is stuck at LRS. By following equaton 3 and 4, huge number of "0" will be created and mapped to the crossbar architecture. Since most of the "0" replaces the SA0 cells, the inference accuracy of the DNN model with SAFs improves .

RMA When SA1:SA0 = 1:1. In previous conditions, the RMA works in such a way that most of weights inside the crossbar array are mapped to either HRS or LRS based on the dominance of the SAF.

But sometimes SA1 and SA0 are equal and happen simultaneously, for example, SA1:SA0 = 1:1. To handle this situation, the RMA uses the same approach of splitting a single weight into two positive weights. But it creates enormous amount of "1" as well as "0" at the same time so that most of the SA1 and SA0 are replaced by those newly mapped "1" and "0". This condition follows the following algorithm.

$$W_a = \begin{cases} W & W > 0, \\ 0 & W < 0, \\ 1 & W = 0, \end{cases} \quad (5) \qquad W_b = \begin{cases} 0 & W > 0, \\ |W| & W < 0, \\ 0 & W = 0, \end{cases} \quad (6)$$

3 RESULT AND DISCUSSION

A physical 40 nm $Ag(Silver)$:a-Si (amorphous Silicon) memristor is manufactured and tested thoroughly. Characteristics of the $Ag(Silver)$:a-Si memristor is incorporated into DNN+NeuroSim platform for evaluations. DNN+NeuroSim is an integrated framework that emulates neural networks (DNN) inference performance on the memristor-based hardware [17]. Here, an 8-layer DNN model VGG8 for CIFAR10 dataset is utilized for the evaluation.

3.1 Weight Distribution After Applying RMA

To enhance the immunity against SAFs when SA1:SA0 = 5:1, after applying the RMA, the initial weight distribution of the VGG8 model, as shown in Fig. 1, is altered into a different shape, as shown in Fig. 2 [16]. Equation 1 and 2 enable this feat. When SA1 is dominant, the algorithm maps 69.96% weights to "1" or HRS. As a result, most of the newly mapped "1" replaces the defective cells that are stuck at HRS (SA1). Similarly, the other 30.04% cells are mapped to sub-1 regions (less than 1 but greater than or equal 0.9). If those sub-1 values are mapped to the SA1 defective cell, the deviation between the mapped weight and the weight from the algorithm is very insignificant. Accordingly, it results in very low accuracy loss in extreme SAF conditions.

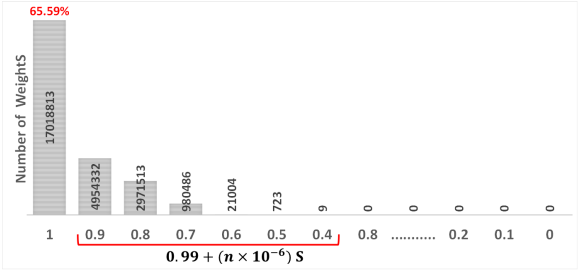

Figure 2: Weight Distribution of VGG8 Model After Applying RMA for SA1:SA0 = 5:1

Similarly, if the SA0 is dominant inside the crossbar, that is SA1:SA0 = 1:5, through equation 3 and 4, the RMA maps 65.71% of the weights to the "0" or LRS, and the rest of the 34.29% weights are also mapped to near zero regions, as shown in Fig. 3. Most of the newly mapped "0" replaces the defective cells that are stuck at LRS (SA0). Therefore, memristors affected by the SA0 act like a defect free device and do not contribute much to the degradation of the inference accuracy of the DNN.

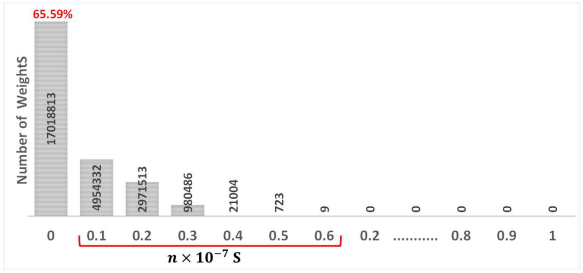

Figure 3: Weight Distribution of VGG8 Model After Applying RMA for SA1:SA0 = 1:5

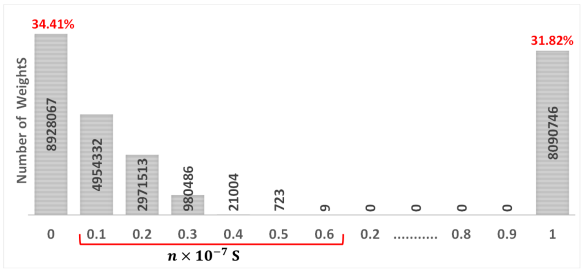

Figure 4: Weight Distribution of VGG8 Model After Applying RMA for SA1:SA0 = 1:1

However, when both parts of the SAF are dominant, equation 5 and 6 assigns a large number of weights to both LRS and HRS. As shown in Fig. 4, 34.41% weights are converted to "0" or LRS and 31.82% weights are converted to "1" or HRS in case of SA1:SA0 = 1:1. The rest of the weights are very small, and most of them almost equal to zero which are not so much affected by the SA0 defects. By mapping enormous weights to "0" or "1", the RMA creates significant immunity against SA1:SA0 = 1:1.

3.2 Accuracy with RMA under Different SAF

Accuracy Restoration When SA1:SA0 = 5:1. The original inference accuracy of the DNN model achieved by the ideal memristor-based neuromorphic computing systems is 90% without any SAFs. To investigate the deteriorating impact of SAF, SAFs from 0.1% to 50% is introduced. As shown in Table. 1, before the RMA is used, the inference accuracy decrease starts as 0.2% SAFs. From 2.5% SAFs, the DNN model becomes completely damaged and provides only 10% accuracy which is like random guessing. The RMA can restore the inference accuracy to 90% when the SAF is less than 10%. The RMA is also super-efficient even with extreme conditions. At 50% SAFs, it achieves 80% accuracy.

Accuracy Restoration When SA1:SA0 = 1:5. Similarly, when SA1:SA0 = 1:5, conventional mapping degrades the inference accuracy quicky. As listed in Table. 1, with as small as 1% SAFs, the DNN model becomes completely damaged and shows only 10% accuracy. The RMA quickly recovers the original inference accuracy (90%) when SAF is smaller than or equal 10%. Table. 1 also explains that, for the RMA with SA1:SA0 = 1:5, the maximum and minimum accuracy improvements are 80% and 72%, respectively.

Table 1: Accuracy Degradation and Restoration with Different SAF Conditions

SAF (SA1 and SA0)	Accuracy No SAF	Accuracy Before RMA			Accuracy After RMA		
		SA1: SA0 = 5:1	SA1: SA0 = 1:5	SA1: SA0 = 1:1	SA1: SA0 = 5:1	SA1: SA0 = 1:5	SA1: SA0 = 1:1
0.10%		90%	88%	90%	90%	90%	90%
0.20%		89%	83%	90%	90%	90%	90%
0.50%		80%	12%	90%	90%	90%	90%
1%		42%	10%	90%	90%	90%	90%
2.50%		10%	10%	90%	90%	90%	90%
5%		10%	10%	88%	90%	90%	89%
7.50%	90%	10%	10%	83%	90%	90%	89%
10%		10%	10%	54%	89%	90%	89%
15%		10%	10%	10%	89%	89%	88%
20%		10%	10%	10%	88%	88%	87%
30%		10%	10%	10%	87%	87%	83%
40%		10%	10%	10%	81%	85%	73%
50%		10%	10%	10%	80%	82%	66%

Accuracy Restoration When SA1:SA0 = 1:1. Unlike the other two conditions, even the conventional mapping can create a relatively strong immunity with low ratio SAFs when SA1:SA0 = 1:1. As shown in Table. 1, the conventional mapping can achieve high accuracy up to 7.5% SAFs without the RMA. As shown in Fig. 1, 31.18% weights of the DNN model are neutral and the overall mean of the eight layers are inclined towards zero. When the conventional mapping deals with SA1:SA0 = 1:1, the circuit nearly reads equal number of "1" and "-1" from the crossbar array based on the resistance level [LRS, HRS]. After the execution of the DNN model, those "1" and "-1" cancel out each other's adverse impact and bring back the mean value closer to zero again. However, the inference accuracy of the conventional mapping drops fast when the SAFs are greater than 7.5%. It even drops to 10% accuracy at 15% SAFs. With the RMA, the DNN model achieves approximately 90% accuracy even with significantly high ratio SAF. According to Table. 1, the RMA can improve the inference accuracy up to 79%. The reason behind this significant improvement is shown in Table 2. Here a practical scenario, SA1:SA0 = 1:1 with 10% SAF, is taken as an example. After the RMA is applied, approximately 6% of defective cells are successfully replaced by the newly mapped "0" (LRS) and "1" (HRS). So, the 10% SAFs acts like a 4% SAFs. Accordingly, as listed in Table. 1, the RMA restores 89% accuracy where conventional mapping provides only 54% accuracy at 10% SAFs under SA1:SA0 = 1:1. Table 1 also shows that conventional mapping achieves 88% accuracy with 5% SAFs which validates the result of achieving 89% accuracy when the RMA deals with 10% SAFs (the visible defect is 4% in Table 2, similar with 5%).

Moreover, the scenario in Table 2 is the worst-case scenario. After the RMA is applied, the range of weight is squeezed within [0,1] instead of [-1,1] and 34.41% weights are mapped to near zero region, as shown in Fig. 4. The RMA not only reduces the weight range but also creates a huge near zero regions that helps compensate the accuracy loss caused by the SA0 defective cells. However, the RMA struggle to achieve very high accuracy with extreme 50% SAF, when SA1:SA0 = 1:1. This is because, when the SAF is 50%, SA1 is 25% and SA0 is 25%. But as shown in Fig. 4, although the RMA maps 34.41% cells to "0" and 31.82% cells to "1", they can not replace all SA1 and SA0 defective cells. It possibly results in relatively low inference accuracy.

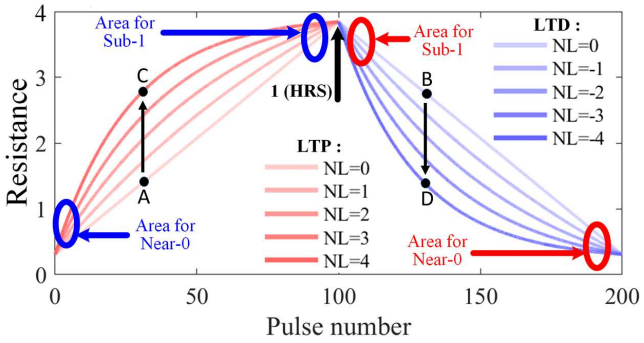

Figure 5: Non-Linearity of Memristor cells.

3.3 Accuracy Restoration with Non-Linearity

Apart from SAFs, memristor is additionally afflicted by non-linearity. The weight of the synapse is represented by the resistance/conductance of the memristor, which must be updated frequently during the training and inference process as specified by learning algorithms. Weight increment (or long-term potentiation, LTP) and decrement (or long-term depression, LTD) should ideally be proportionate to the number of writing pulses [6]. However, physical restrictions such as inherent drift and diffusion dynamics of the ions/vacancies cause the inaccurate weight updating with respect to the input stimulus pulse(s) [8, 22]. Ideally, in the weight updating process, the change in the resistance of an ideal synapse device is proportional to number of stimulus pulses. In Fig. 5, the curves (dark) represent the actual resistance value of a memristor device with respect to the number of input pulses where the pulses possess the same duty cycle and the same amplitude, and the straight line (light) represents the hypothetical resistance value of the ideal case. For instance, as shown in Fig. 5, with LTP = 0 and LTD = 0, two ideal memristors produce two resistive states A and B, respectively.

However, with strong nonlinearity LTP = 4, LTD = -4, an abrupt incline and decline of the resistive state is obtained, which are labeled as C and D, respectively. This inaccurate weight updating directly impact the overall performance of the DNN model.

The nonlinearity of LTP = 1.75 and LTP = -1.46 is taken into account in all of the results from Tables 1 and 2. However, the conventional mapping totally fails with the inclusion of very high nonlinearity. The RMA is efficient in restoring the high accuracy even with high nonlinearity. As listed in Table 3, when SAF is 1% for SA1:SA0 = 1:5, 1:1, and 5:1, the RMA recovers 70% inference accuracy which was 10% before. Similarly when SAF is 20%, the RMA achieves over 60% accuracy.

This is because the RMA maps majority of the weights at the LRS or HRS where the influence of the non-linearly is absent, as shown in Fig. 5. Hence, the nonlinearity cannot cause any negative impact. As shown in Figs. 2, 3, and 4, when SA1:SA0 = 5:1, 99% weights are mapped to "1" or HRS and sub-1 region; when SA1:SA0 = 1:5, 99% weights are mapped to "0" or LRS and sub-0 region; when SA1:SA0 = 1:1, 99% weights are mapped to HRS and LRS; they are all not affected by the non-linearity. Since in each case, the tendency of the RMA is to map weights to the HRS or LRS are almost always equal about 65% (as shown in Figs. 2, 3, and 4), the RMA has a similar effectiveness for three conditions SA1:SA0 = 1:5, 1:1, and 5:1. Accordingly, Table 3 listed a similar accuracy restoration after applying the RMA for three conditions.

Table 2: Comparison of Actual Defect and Visible Defect After Applying RMA When SAF = 10%

Number of Iterations	Actual Defect	Visible Defect
1	10%	4.02%
2		3.98%
3		4.02%
4		4.03%
5		4.04%
6		3.99%
7		4.05%
8		4.00%
9		4.00%
10		4.01%
11		4.03%
12		4.03%

3.4 Retention

Retention is defined as the ability of the memristor device to retain its programmed state over a long period of time [4]. Typically the retention ability of a memristor is more than 10 years at 85 degrees Celsius. As shown in Table 4, four conductance drift scenarios have been discussed for the retention analyzed: Drift to HRS, Drift to LRS, Drift to Middle, and Random Drift.

Here 10% SAF along with different drift has been considered in Table 4. The RMA can successfully restored the high inference accuracy even with the different drift conditions. The reason is that we split a single weight into two numbers and store it in two memristors. In some cases, the two weights are affected by the same amount of drifts, and therefore after the subtraction, the same difference is obtained before/after drift. Finally, the DNN can get the desired weight during execution.

Table 3: Accuracy Restoration Using Adaptive Mapping Method in Presence of Significant Non-Linearity

SAF	Accuracy Before RMA (LTP = 4, LTD = -4)			Accuracy After RMA (LTP = 4, LTD = -4)		
	5:1	1:5	1:1	5:1	1:5	1:1
0.1%	61%	56%	71%	71%	71%	71%
1%	10%	10%	70%	70%	70%	70%
2.5%	10%	10%	65%	70%	69%	65%
20%	10%	10%	10%	64%	67%	60%
50%	10%	10%	10%	50%	50%	24%

Table 4: Impact of Drift on RMA

Drift	SA1:SA0 = 5:1	SA1:SA0 = 1:5	SA1:SA0 = 1:1
Drift to HRS	89%	90%	90%
Drift to LRS	89%	90%	88%
Drift to Middle	90%	90%	90%
Random Drift	89%	89%	89%

3.5 Chip Area Estimation

The memristor based neuromorphic computing chip is made up of a number of tiles, a global buffer, neural functional computation units such as accumulating units, activation units, pooling units, as well as computation units for weight gradient. In each tile, there are several processing elements (PEs), tile buffers for loading neural activations, accumulation modules for adding partial sums from PEs and output buffers. The total size of a memristor based DNN chip is shown in Table 6.

The total chip area with RMA is 0.38% larger than the DNN without the RMA. It can be negligible, considering the great contribution of the RMA on the accuracy and immunity to SAFs.

Table 5: State-of-The-Art

State-of-The-Art	Parameters				
	No Intricate Algorithm	No Complex Read Circuit	No Separate Customization	Consideration of all Possible SAF Ratio	High Accuracy Restoration on all Possible SAF Ratio
[16]	√	√	√	×	×
[15]	×	×	×	×	×
[3]	×	×	×	×	×
[20]	×	√	×	×	×
[18]	×	×	×	×	×
[11]	×	√	√	×	×
[21]	×	√	×	×	×
[1]	×	√	×	×	×
[10]	×	√	×	×	×
[19]	√	√	×	×	×
This Work	√	√	√	√	√

4 COMPARISON WITH STATE-OF-THE-ART

As shown in Table 5, our proposed method offers some advantages over the state-of-the-art. So far, most of the SAF handling

Table 6: Area Comparison Between Before and After Reconfigurable Mapping Algorithm (RMA)

Items	Before RMA	After RMA
Total Compute-In-Memory (memristor) array	$3.65072 \times 10^5 \mu m^2$	$7.30144 \times 10^5 \mu m^2$
Total IC Area on chip (Global and Tile/PE local)	$1.20372 \times 10^7 \mu m^2$	$1.20372 \times 10^7 \mu m^2$
Total ADC Area on chip	$5.40220 \times 10^7 \mu m^2$	$5.40220 \times 10^7 \mu m^2$
Total Accumulation Circuits on chip (Adders, shift Adds accumulation units)	$1.05098 \times 10^7 \mu m^2$	$1.05098 \times 10^7 \mu m^2$
Other Peripheries (decoders, mux, switch matrix, buffers, pooling, and activation units)	$9.57562 \times 10^7 \mu m^2$	$9.57562 \times 10^7 \mu m^2$
Weight Gradient Calculation	$9.66804 \times 10^6 \mu m^2$	$9.66804 \times 10^6 \mu m^2$
Differential Reading Circuit (Op amp based subtractors)	0	$3.01333 \times 10^2 \mu m^2$
Total Chip Area	$9.61777 \times 10^7 \mu m^2$	$10.82152 \times 10^7 \mu m^2$ (0.38% Overhead)

approaches develop an intricate algorithm to determine the significant weights first. Then a complex read circuit identifies SAFs free regions for mapping those significant weights. However, These approaches cause a large hardware and software overhead. The RMA can be used as a ubiquitous solution to avoid all these complexities.

5 CONCLUSION

High integrated density and simple crossbar architecture makes memristor suitable for the implementation of large and complex DNN model in neuromorphic computing systems. But unavoidable SAF defects impede its commercial success, because the inference accuracy drop is inevitable. In this paper, the RMA is proposed to deal with such accuracy degradation. The experiment results show that the RMA can restore the interference accuracy to 90% when the SAF is less than or equal to 7.5%/10%/2.5% at SA1:SA0 = 5:1/1:5/1:1. Even in some extreme cases, for example SAF = 50%, the RMA is also effective and achieves the accuracy up to 80%/82%/66% at SA1:SA0 = 5:1/1:5/1:1. Finally, as compared with state-of-the-art, our proposed method implies the superiority.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under Grant 2218046, Grant 1953544, and Grant 1855646.

REFERENCES

[1] Gouranga Charan, Abinash Mohanty, Xiaocong Du, Gokul Krishnan, Rajiv V Joshi, and Yu Cao. 2020. Accurate inference with inaccurate rram devices: A joint algorithm-design solution. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits* 6, 1 (2020), 27–35. https://doi.org/10.1109/JXCDC.2020.2987605

[2] Ching-Yi Chen, Hsiu-Chuan Shih, Cheng-Wen Wu, Chih-He Lin, Pi-Feng Chiu, Shyh-Shyuan Sheu, and Frederick T Chen. 2014. RRAM defect modeling and failure analysis based on march test and a novel squeeze-search scheme. *IEEE Trans. Comput.* 64, 1 (2014), 180–190. https://doi.org/10.1109/TC.2014.12

[3] Lerong Chen, Jiawen Li, Yiran Chen, Qiuping Deng, Jiyuan Shen, Xiaoyao Liang, and Li Jiang. 2017. Accelerator-friendly neural-network training: Learning variations and defects in RRAM crossbar. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 19–24. https://doi.org/10.23919/DATE.2017.7926952

[4] Pai-Yu Chen and Shimeng Yu. 2018. Reliability perspective of resistive synaptic devices on the neuromorphic system performance. In *IEEE International Reliability Physics Symposium (IRPS)*. 5C–4. https://doi.org/10.1109/IRPS.2018.8353615

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint* (2018). https://doi.org/10.48550/arXiv.1810.04805

[6] Jingyan Fu, Zhiheng Liao, Na Gong, and Jinhui Wang. 2019. Mitigating nonlinear effect of memristive synaptic device for neuromorphic computing. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 9, 2 (2019), 377–387. https://doi.org/10.1109/JETCAS.2019.2910749

[7] Jingyan Fu, Zhiheng Liao, Jianqing Liu, Scott C Smith, and Jinhui Wang. 2020. Memristor-Based Variation-Enabled Differentially Private Learning Systems for Edge Computing in IoT. *IEEE Internet of Things Journal* 8, 12 (2020), 9672–9682. https://doi.org/10.1109/JIOT.2020.3023623

[8] Jingyan Fu, Zhiheng Liao, and Jinhui Wang. 2022. Level Scaling and Pulse Regulating to Mitigate the Impact of the Cycle-to-Cycle Variation in Memristor-Based Edge AI System. *IEEE Transactions on Electron Devices* 69, 4 (2022), 1752–1762. https://doi.org/10.1109/TED.2022.3146801

[9] Bogdan Govoreanu, Gouri Sankar Kar, YY Chen, Vasile Paraschiv, Stefan Kubicek, Andrea Fantini, IP Radu, Ludovic Goux, Sergiu Clima, Robin Degraeve, et al. 2011. 10×10nm 2 Hf/HfO_x crossbar resistive RAM with excellent performance, reliability and low-energy operation. In *International Electron Devices Meeting*. 31.6.1–31.6.4. https://doi.org/10.1109/IEDM.2011.6131652

[10] Zhezhi He, Jie Lin, Rickard Ewetz, Jiann-Shiun Yuan, and Deliang Fan. 2019. Noise injection adaption: End-to-end ReRAM crossbar non-ideal effect adaption for neural network mapping. In *56th Annual Design Automation Conference*. 1–6. https://doi.org/10.1145/3316781.3317870

[11] Giju Jung, Mohammed Fouda, Sugil Lee, Jongeun Lee, Ahmed Eltawil, and Fadi Kurdahi. 2021. Cost-and dataset-free stuck-at fault mitigation for ReRAM-based deep learning accelerators. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 1733–1738. https://doi.org/10.23919/DATE51398.2021.9474226

[12] Kuk-Hwan Kim, Sung Hyun Jo, Siddharth Gaba, and Wei Lu. 2010. Nanoscale resistive memory with intrinsic diode characteristics and long endurance. *Applied Physics Letters* 96, 5 (2010), 05310.1–53106.3. https://doi.org/10.1063/1.3294625

[13] Shahar Kvatinsky, Eby G Friedman, Avinoam Kolodny, and Uri C Weiser. 2012. TEAM: Threshold adaptive memristor model. *IEEE transactions on circuits and systems I: regular papers* 60, 1 (2012), 211–221. https://doi.org/10.1109/TCSI.2012.2215714

[14] Zhiheng Liao, Jingyan Fu, and Jinhui Wang. 2021. Ameliorate Performance of Memristor-Based ANNs in Edge Computing. *IEEE Trans. Comput.* 70, 8 (2021), 1299–1310. https://doi.org/10.1109/TC.2021.3081985

[15] Chenchen Liu, Miao Hu, John Paul Strachan, and Hai Li. 2017. Rescuing memristor-based neuromorphic design with high defects. In *54th ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6. https://doi.org/10.1145/3061639.3062310

[16] Md Oli-Uz-Zaman, Saleh Ahmad Khan, Geng Yuan, Zhiheng Liao, Jingyan Fu, Caiwen Ding, Yanzhi Wang, and Jinhui Wang. 2022. Mapping Transformation Enabled High-Performance and Low-Energy Memristor-Based DNNs. *Journal of Low Power Electronics and Applications* 12, 1 (2022), 10–24. https://doi.org/10.3390/jlpea12010010

[17] Xiaochen Peng, Shanshi Huang, Hongwu Jiang, Anni Lu, and Shimeng Yu. 2020. DNN+NeuroSim V2.0: An end-to-end benchmarking framework for compute-in-memory accelerators for on-chip training. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 40, 11 (2020), 2306–2319. https://doi.org/10.1109/TCAD.2020.3043731

[18] Lixue Xia, Mengyun Liu, Xuefei Ning, Krishnendu Chakrabarty, and Yu Wang. 2017. Fault-tolerant training with on-line fault detection for RRAM-based neural computing systems. In *54th Annual Design Automation Conference*. 1–6. https://doi.org/10.1145/3061639.3062248

[19] Injune Yeo, Myonglae Chu, Sang-Gyun Gi, Hyunsang Hwang, and Byung-Geun Lee. 2019. Stuck-at-fault tolerant schemes for memristor crossbar array-based neural networks. *IEEE Transactions on Electron Devices* 66, 7 (2019), 2937–2945. https://doi.org/10.1109/TED.2019.2914460

[20] Baogang Zhang, Necati Uysal, Deliang Fan, and Rickard Ewetz. 2019. Handling stuck-at-fault defects using matrix transformation for robust inference of dnns. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 10 (2019), 2448–2460. https://doi.org/10.1109/TCAD.2019.2944582

[21] Jiangwei Zhang, Donald Kline, Liang Fang, Rami Melhem, and Alex K Jones. 2017. Dynamic partitioning to mitigate stuck-at faults in emerging memories. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 651–658. https://doi.org/10.1109/ICCAD.2017.8203839

[22] Mohammed A Zidan, John Paul Strachan, and Wei D Lu. 2018. The future of electronics based on memristive systems. *Nature electronics* 1, 1 (2018), 22–29. https://doi.org/10.1038/s41928-017-0006-8