UNIVERSITY OF CALIFORNIA

Santa Barbara

Novel Circuits and Systems with Analog-Grade Memories

A dissertation submitted in partial satisfaction of the

requirements for the degree Doctor of Philosophy

in Electrical and Computer Engineering

by

M. Reza Mahmoodi

Committee in charge:

Professor Dmitri Strukov, Chair

Professor Tim Sherwood

Professor James Buckwalter

Professor Li-C Wang

June 2021

The dissertation of M. Reza Mahmoodi is approved.

_____

Tim Sherwood

_____

James Buckwalter

_____

Li-C Wang

_____

Dmitri Strukov, Committee Chair

March 2021

Novel Circuits and Systems with Analog-Grade Memories

By

M. Reza Mahmoodi

Dedicated to you.

ACKNOWLEDGEMENTS

There are no words to truly describe the depth of my gratitude toward my advisor, Dmitri Strukov. Thanks for the unlimited support, guidance, supervision, and inspiration that still wakes me up early in the morning and keeps me up late at night. Thank you for setting an example of excellence as a researcher, academic, and human being, and thank you for taking a chance on me. Your unapproachable high standard and outstanding work ethic will drive me forever. I owe you big time.

I thank my friend, colleague, wife, and partner in crime, Zahra Fahimi, who never deserted me in this expatriate life. Thanks for pushing to the limits and for keeping up with my constant nonsense philosophies. Many thanks also go to the past and present labmates, including Dr. Hussein Nili, Dr. Mohammad Bavandpour, Dr. Michael Klachko, Dr. Xinjie Guo, Dr. Adrien Vincnet, and Dr. Mirko Prezioso. I learned a lot from you guys and inherited a fortune of experience and information. Thank you for creating a nutritious environment in the lab. I also appreciate the brilliant collaborators I was lucky to work with: the inspiring Prof. Lanza, Dr. Omid Kavehi, Prof. V. Polishchuk, my friends in Applied Materials Inc., M. Graziano, and Irina Kataeva.

I am also so thankful to DARPA, BAE, NSF, SRC, Applied Materials, and all the funding agencies which financially supported me during these years. Thank you, Brook and Shan Byers, and thank you, IEE. Your support made a lot of difference.

I thank my parents, Ezollah and Faterme; my sisters, Safien and Batoul; and my brothers, Mohsen, Alireza, and Saeed. I am everything I am because you loved me.

VITA OF M. REZA MAHMOODI
March 2021

Education

Bachelor of Science in Electrical Engineering, Isfahan University of Technology, 2012.
Master of Science in Electronics, Isfahan University of Technology, 2015.
Doctor of Philosophy in Electronics, University of California, Santa Barbara, March 2021 (expected).

Selected Publications[1]

M. R. Mahmoodi, Prezioso, D.B. Strukov. Versatile stochastic dot product circuits based on nonvolatile memories for high performance neurocomputing and neurooptimization, *Nature Communications* **4**, 1-10, (2019).

M. Prezioso*, **M. R. Mahmoodi***, *et al.* Spike-timing-dependent plasticity learning of coincidence detection with passively integrated memristive circuits, *Nature Communications* **9**, 1-8 (2018).

H. Nili, G. C. Adam, B. Hoskins, M. Prezioso, J. Kim, **M. R. Mahmoodi**, F. Merrikh Bayat, O. Kavehei, and D. B. Strukov. Hardware-intrinsic security primitives enabled by analogue state and nonlinear conductance variations in integrated memristors, *Nature Electronics* **1**, 197-202 (2018).

S. Chen, **M. R. Mahmoodi**, *et al.* Wafer-scale integration of two-dimensional materials in high-density memristive crossbar arrays for artificial neural networks, *Nature Electronics* **3**, 638-645 (2020).

H. Kim*, **M. R. Mahmoodi***, H. Nili*, D. Strukov. 4K-memristor analog-grade passive crossbar circuit, *arXiv preprint arXiv:1906.12045* (2019).

M. R. Mahmoodi*, Z. Fahimi*, H. Nili, Valentin Polishchuk, D. B. Strukov. Combinatorial Optimization by Weight Annealing in Memristive Hopfield Networks, *Nature Scientific Reports*, (2021).

Awards

IEE Excellence in Research Fellowship Recipient (2018-2019) .

[1] See https://scholar.google.com/citations?user=ATd-RqUAAAAJ&hl=en&oi=ao for a full list of publications.

ABSTRACT

Novel Circuits and Systems with Analog-Grade Memories

by

M. Reza Mahmoodi

The neural computation field had finally delivered on its promises in 2013 when the University of Toronto group reported a deep neural network that outperformed other machine learning approaches in image classification accuracy. That breakthrough was not due to algorithmic advances but rather the availability of high-performance graphical processors that enabled large-scale neural network modeling. Since then, the biologically-inspired neural network algorithms have become state-of-the-art approaches in many artificial intelligence tasks, and the future progress in this field hinges on even more powerful hardware. Such hardware, however, is unlikely to be implemented with the conventional digital circuit technology, whose performance seems to be saturating due to the faltering Moore's law.

On the other hand, further opportunities are presented by neuromorphic hardware that mimics critical features of biological neural networks, most importantly analog in-memory computing, in an attempt to match their energy-efficiency. Most importantly, neuromorphic hardware takes advantage of the physical-level analog implementation of vector-by-matrix multiplication (VMM), which is the most frequent operation in any neural network. The key component of such a circuit is a nanodevice with adjustable conductance —essentially an analog nonvolatile memory—used at each crosspoint of a crossbar array and mimicking the biological synapse. Prior work showed that analog VMM circuits based on redesigned eFlash

memories and metal-oxide memristors, the most promising analog memory device technologies for neuromorphic computing, are much more energy-efficient as compared to the digital counterpart implemented in similar process node and performing a similar function.

The main goal of this dissertation is to advance neuromorphic circuits based on memristors and eFlash memories on several fronts. The first part of the thesis is devoted to improving functional and physical performance of analog-domain vector-by-matrix multiplication with a specific focus on neuromorphic inference applications, including the development of novel programming algorithms, mitigation approaches for various device and circuit non-idealities, and design of efficient peripheral circuits. For example, we use novel programming algorithms to experimentally demonstrate <4% relative tuning error in a 64x64 passively integrated crossbar circuit despite significant variations, with 25% normalized standard deviation in device I-V characteristics. The developed post-fabrication methods for mitigating IR drops, I-V static nonlinearity, and device variations enable software-equivalent accuracy for the large-scale neural networks for the studied memristor technology. The efficacy of novel peripheral circuits is verified via SPICE modeling, which shows, e.g., POp/J-scale energy-efficiency for current-mode 55-nm NOR-flash memory circuits. The section is concluded with the discussion of our ongoing work on the design and fabrication of several large-scale neuromorphic chips.

The second part of this thesis extends the work on analog VMM circuits to enable the implementation of more advanced probabilistic neuromorphic hardware, which is especially effective in solving combinatorial optimization problems. By operating the previously developed analog VMM circuit in a lower signal-to-noise-ratio regime, we achieve stochastic VMM functionality and utilize such circuits to prototype small-scale restricted Boltzmann

machine and Hopfield neural network with runtime-controlled effective temperature. Furthermore, we suggest several novel hardware-friendly annealing approaches and successfully verify them by solving experimentally typical combinatorial optimization problems.

The last part of this dissertation is devoted to hardware security primitives, such as physically unclonable functions and true random number generators. At the core of our idea are analog circuits based on metal-oxide memristors and eFlash memories, which are very similar to analog VMMs developed for neuromorphic computing. The main difference is that memory device non-idealities, e.g., randomness in tuning and memory *I-V* variations, are utilized as a rich source of static entropy, which is essential for implementing hardware security primitives. We developed three architectures - RX-PUF and VR-PUF that avoid the need for conductance tuning procedure in previously proposed memristor-based PUFs, and ChipSecure, which exploits variations in leakage current, subthreshold slope, nonlinearity, and stochastic tuning error in eFlash memory arrays to create a unique digital fingerprint. The key novelties of the proposed designs include enormous challenge-response pairs to enable strong PUF properties and a low-overhead key-booking scheme to dramatically improve the PUF reliability across a wide temperature range of operation. The analysis of the measured data in all our PUF demonstrations shows strong resilience against machine learning attacks.

TABLE OF CONTENTS

1. Introduction

1.1. Motivation

The field of neural computation has had its highs and lows in the last three decades. Still, the huge milestone happened only recently: high-performance graphical processors and the availability of massive labeled datasets enabled the development of deep neural networks (DNNs) that exhibit superior performance in various pattern recognition tasks [1]. The progress is still ongoing: breakthroughs in algorithms are fueled by increasingly powerful hardware accelerators, processors, and aggressively scaled custom digital circuits. Unsurprisingly, further progress hinges on the use of more efficient hardware as well. Biological systems have played an undeniable role in this progress and have served as a massive inspiration for building intelligent systems. Yet, even advanced digital neural networks fail to compete with them in terms of performance. The human brain outperforms artificial systems in many tasks, e.g., it can recognize an intricate object in an image faster while consuming orders of magnitude less energy [2]. The main reason stems from the fact that using digital operations to mimic noisy redundant biological systems is inherently natural. In other words, this efficiency gap originates from the use of elementary physical phenomena as computational primitives and the analog representation of information in biosystems [3]—the observation that initiated the field of neuromorphic engineering [4].

A tremendous body of research predicts analog neuromorphic networks could bridge the gap between artificial and biological prototypes [5-7] and offer comparable areal density to biological prototypes at a better processing time. The background for this improvement is the physical level implementation of the vector-by-matrix multiplication (VMM), the frequent

operation in neuromorphic networks, and the efficient realization of an analog synapse capable of both holding a learnable parameter (or weight) and performing useful computations.

One key challenge in achieving the promised excellent performance is developing a technology that offers reliable analog-grade programmability and can be easily integrated with CMOS. In general, a promising synaptic device should offer several primary features: 1) long-term retention, that is, the device should hold the stored information for a reasonable time period, 2) high endurance, that is, the device should be reprogrammable for a reasonable amount of time, 3) analog-storage, that is, the device should be able to store the parameters with reasonable accuracy, 4) low-power operation, that is, the energy consumption needed to access and perform the computation on the device should be reasonably low, 5) compact footprint, that is, the device should be scalable and dense to ensure a large number of synapses can be integrated within a single chip with minimum parasitics and high-speed access time.

In every neuromorphic network, there are many infrequent operations that standalone synaptic devices cannot implement, and in some cases, they are very inefficient to do so. Fortunately, the mature CMOS technology is extremely flexible and can efficiently implement various functionalities. The second challenge is designing efficient peripheral circuits in CMOS technology that perform and implement less frequent and sparser functionalities of these systems.

Besides accelerating deep neural networks, addressing these challenges would be a significant milestone in implementing other neuromorphic tasks, such as Boltzmann machines [8], spiking neural networks [9,10], Hopfield neural networks [9,11,12], reinforcement learning [13], etc., all of which have been proven to be very efficient when implemented with

analog-grade nonvolatile memories (NVMs). Further, integrated circuits based on analog memories are promising for designing very efficient security primitives such as physically unclonable functions [13] and random number generators [15], as they offer extra sources of randomness and functionalities in far better densities than purely-CMOS circuits.

The recent development of continuous-state nonvolatile memory synapses is perhaps a milestone that paves the way for efficiently building these systems [16]. The most notable device candidates that excel in primary features are metal-oxide passive memristors [17] and redesigned eFlash memories. In this Ph.D. thesis, we focus on developing these technologies and harnessing them for designing novel circuits and systems targeting neuromorphic applications and hardware security. Our study spans different design levels, from developing nonvolatile memory technologies, addressing their reliability issues, studying their scaling prospects, to designing energy-efficient and high-speed peripheral circuits and using them for building large-scale integrated neuromorphic networks and hardware security primitives.

The remainder of this chapter focuses on background works. Chapter 2 reviews various devices and topologies for building the basic building block of neuromorphic computing systems, i.e., VMMs. Then, we report our results on the successful fabrication of large-scale passively-integrated memristive crossbars. The study is followed by an in-depth analysis of the role of uniformity, IR drop, and nonlinearity in 0T1R (transistorless) memristive circuits. Then, we propose two circuit techniques for developing a low-power peripheral circuit, leading to a high-speed and energy-efficient current-mode VMM design. Finally, we introduce three large-scale integrated inference engines: a 6-layer convolutional neural network and a 3-layer multilayer perceptron (MLP) based on eFlash memories in 55 nm

CMOS, and a massive general-purpose accelerator engine based on 1T1R (1 transistor + 1 resistor) memories.

Chapter 3 is devoted to neuromorphic computing systems and circuits, in which we show the efficient design of restricted Boltzmann machines, Hopfield neural networks, and various annealing techniques for solving combinatorial optimization problems. Finally, in chapter 4, we discuss multiple successful demonstrations of strong physical unclonable function circuits and hardware security primitives based on resistive memories and redesigned eFlash arrays.

1.2. Background and Significance

1.2.1. Devices

The principles of analog computing date back to half a century ago [3,18,19]. Up until recently, such devices were implemented mostly as "synaptic transistors" [7], fabricated in some double-poly CMOS technologies. Several sophisticated analog computing systems were proposed using these devices [20]. However, these devices have relatively large areas, leading to sparse implementation and large interconnect parasitics. An efficient realization of analog computing is enabled only recently, with the emergence of novel analog-grade nonvolatile memories that implement VMM operation using the fundamental Ohm and Kirchhoff laws. Among different candidates [21], the resistive switching memories, including phase change [22], conductive bridge memories [23], and metal-oxide memristors [24,25], are promising candidates. These synaptic devices are implemented in 0T1R [24] and 1T1R [25] topologies. In comparison with 1T1R, the fabrication and upscaling of 0T1R arrays are more challenging, in part due to the large uniformity requirements in 0T1R arrays, which explains few experimental demonstrations of these circuits. In a uniform 0T1R technology, the devices

have a tight distribution of switching thresholds, which leads to minimum half-select disturbance in the crossbar, facilitating individual tuning of the devices. In 1T1R technology, this requirement is waived due to the presence of the selector in each memory cell. However, this comes at the cost of dramatic density reduction. In addition, the 0T1R technology offers 3-dimensional integration [26], which is not feasible in 1T1R memories.

The high prospective integration density of passive memristive crossbar circuits, enabled by both aggressive lateral feature scaling and vertical monolithic integration, would be essential for hardware implementations of large neural network models, such as those used for the end-to-end automatic speech recognition, natural language translation, and text summarization, on a single chip without having to perform very energy-taxing and slow data transfer with the off-chip memory. For example, the largest multilingual neural model for automatic translation among seven common languages contains 640 million parameters [27]. The functional performance of the transformer networks, the state-of-the-art models for text summarization, dramatically improves with the scale of the network, e.g., almost linearly improving when increasing the number of parameters in GPT-2 model from few hundreds to ten billion [28]. Furthermore, mixture-of-expert networks with up to 137 billion parameters have been recently suggested to improve the functional performance of language modeling [29].

Storing that many parameters on-chip could hardly be accommodated with planar embedded memory technologies. Earlier work showed that memory devices in the general-purpose neuromorphic computing chips could occupy up to 25% of the total area, with the remaining area devoted to memory array peripheral circuits and other functions [16]. The memory efficiency would be naturally higher, and hence memory density is more important

in more-specialized circuits, e.g., large-scale models that do not rely on weight sharing. In fact, our crude estimates show that even with largely suboptimal (higher) cell currents, memory efficiency is above 10% for neuromorphic inference accelerators with 100M+ weights. At the same time, though the complexity of the mentioned above large-scale neural networks might reduce with further improvements in algorithms, it is clear that extremely large models will still be useful. This can be indirectly evidenced by the complexity of the human brain, which, with its ~10^{15} synapses, can serve as a proxy for the complexity of the future highly cognitive neuromorphic systems [30].

While the most promising synaptic device option for realizing neuromorphic systems are analog-grade (i.e., multi-bit) devices, low-precision (e.g., binary weight) neural network models have also received significant attention [31,32,33]. However, it seems that understanding and dealing with the impact of reduced weight and computing precisions is still a very active area of research. For example, though little or no loss in accuracy can be achieved when using binary weights for some of the earlier (very redundant) deep convolutional networks, such as AlexNet or VGG, 4 to 8 bits of precision for both weights and activations might be necessary for the most advanced image classifiers [34]. A related observation is that the accuracy loss can often be recovered by increasing the network depth or width [35,36], which, however, naturally results in decreased physical performance. Therefore, the prospects for lowering precision in the neural network, which might enable using simpler binary resistive random access memory (ReRAM) devices, can only be understood by considering both functional and physical performances at the system level [37].

The main advantages of using passively integrated metal-oxide memristors [24,26,38] are their superior density and lower fabrication cost [29]. In fact, due to excellent scaling

prospects and analog properties, vertically integrated ReRAMs might challenge much slower 3D NAND memories in effective density to enable human-brain-scale integrated electronics. The progress in developing passive ReRAM technology is still ongoing, and the technology is still in need of improvement.

Another perfect candidate is commercially available eFlash memory technology from Silicon Storage Technology (SST) Inc. This technology has been successfully scaled down to 28 nm and is embedded in many standard CMOS foundries and processes [39]. However, the baseline floating-gate technology is designed for digital NOR flash memory applications and does not allow setting a precise analog state of each cell, necessary for analog applications. Recent work from our group has shown that redesigning the routings in the cells mitigates the half-select disturbance in these arrays and allows their high-precision tuning [40,41]. Such redesigning has increased the cell area by ~×3 but has enabled analog tuning capability in the devices. The main advantage of this technology is their mature, CMOS-compatible fabrication technology and their high-precision capability, which is offered in a density of only ~100 F^2/cell.

Chapter 2 studies and qualitatively compares various topologies and device candidates for implementing dot-product operation in the mixed-signal domain. Then, we demonstrate a 64×64 passive crossbar circuit with ~ 99% functional nonvolatile metal-oxide memristors featuring etch-down patterning and low-temperature budget (hence conducive to vertical monolithic integration) based on a foundry-compatible fabrication process. Most importantly, the achieved device uniformity, < 26% coefficient of variance in memristor switching voltages, is sufficient for programming a 4K-pixel gray-scale pattern with a smaller than 4% relative tuning error on average. Chapter 2 also reports the results of an in-depth analysis of

uniformity in passive memristive circuits and investigates how it impacts the computing accuracy of analog memristive circuits.

1.2.2. Circuits and Systems

a) Neurooptimization and Annealing Hardware

The enormous computational power required to solve large-scale optimization problems poses a great challenge for their efficient implementation. Hence, hardware accelerators, e.g., based on superconductors [45], CMOS circuits [46], nanomagnetic devices [47], and photonic technologies [48]. These circuits employ annealing techniques such as CMOS annealing [46], quantum [45], stochastic [49], and chaotic annealing [50] to boost performance.

The stochastic dot-product computation is the most common operation performed during inference and training in the Boltzmann machine and simulated annealing. Hence, its efficient hardware realization is of utmost importance. Even with a relatively large synapse to neuron ratio (~1,000) and deterministic dot-product functionality, the neuron circuitry might constitute a substantial part of the neuromorphic inference systems. Because of such concerns, purely CMOS implementations, see, e.g., CMOS probabilistic gates [51] and CMOS-based Ising chip for combinatorial optimization problems [46], may not be very practicable. CMOS annealing may require a random number generator, reports low area efficiency (1.225 μm^2/4spins in 40 nm), and has many constraints (e.g., weights must be binary). Quantum computing systems, on the other hand, are yet to be explored more, e.g., to achieve scalability, overcome the challenges of operation at room temperature and at the presence of environmental noise.

Recent progress in developing analog-grade NVMs has created exciting opportunities for the experimental realization of Hopfield networks and annealing machines.

This, in part, stems from the fact that the implementation overhead of stochastic functionality might be less of a problem for some memory devices, in which switching between memory states is inherently stochastic. Ferromagnetic [52,53], phase-change [54,55], ionic [56,57] and thermally-driven metal-oxide [12], and solid-state electrolyte devices [58,59] are the recent proposed candidates to implement stochastic functionality. Ref. [60] uses discrete Pt/TiO_{2-x}/Pt memristive devices to implement a small-scale 4-bit data converter with the Hopfield model. Ref. [61] implements a 3-bit associative memory using digital HfO_2 memristors. In Ref. 12, simulation results demonstrate the effectiveness of using the inherent chaos in sub-100 nm NbO_2 memristors to implement simulated annealing within Hopfield networks. Ref. [8] shows an 18-node restricted Boltzmann machine (generalized Hopfield network) and a 4-node graph partitioning problem based on a versatile stochastic dot-product engine using TiO_2 memristive crossbars. Ref. [62] uses Y-flash memories to implement a 3-bit associative memory.

Unfortunately, many of such devices come with other severe challenges. For instance, an efficient implementation of large-scale dot-product computation is a major challenge for magnetic devices. The hybrid option of combining magnetic stochastic neurons with the already mentioned mixed-signal dot-products is not appealing because the interface typically compromises an extreme energy efficiency of spin-based computing with charge-based devices. The technology of magnetic devices is also relatively immature, judging by very few (and rather low-complexity) experimental demonstrations [60,61]. The biggest challenge for the remaining devices would be low switching endurance and cycle-to-cycle and device-to-device variations in switching characteristics.

Chapter 3 discusses our approach for addressing these limitations through novel circuits and algorithms for implementing restricted Boltzmann machine (RBM) and Hopfield

networks. We experimentally demonstrate hardware implementation of a 10×12 RBM, simulated annealing, chaotic annealing, and novel weight annealing, using crossbars of analog NVMs.

b) MLP and Deep Neural Network Accelerators

The vast majority of the proposed neuromorphic accelerators from industry and academia are digital [63-65]—see also extensive review in Ref. [66]. The most natural approaches, however, are based on analog and mixed-signal circuits. The majority of accelerators based on analog memories have been theoretical (for a list of these architectures and their pros and cons, see the supplementary information in our recent work [67]). Several works also utilize binary ReRAM for building neural network accelerators. Ref. [68] uses a 1 Mb 1T1R array in a 65 nm process to perform the binary dot-product operation in <16 ns. Panasonic [69] also developed a 2 Mb 1T1R ReRAM array for implementing a huge MLP, which surprisingly achieves only ~94% on MNIST.

Few works also use the more interesting analog resistive memories. The first perceptron was implemented based on a very uniform 12×12 0T1R crossbar array to classify 3×3 binary images [24]. The second generation of UCSB's crossbars were 20×20, which were used to implement an MLP network [17]. Ref. [25] demonstrates in-situ training techniques of perceptron networks based on very sparse 128×64 1T1R arrays. The largest ReRAM-based demo is presented in [69], which includes board-level integration of 8 128×16 100 μm^2 1T1R devices, and is used for the implementation of a convolutional neural network.

A milestone in the field of neural computation also happened recently, when an MLP image classifier with <1 μs inference time and <20 nJ inference energy was demonstrated using redesigned 180 nm embedded eFlash memories [70]. This chip offers 10^6 better energy-

delay in comparison with IBM's TrueNorth [71]. More importantly, the results have been reproducible, reliable, and temperature insensitive.

Section 2.7 discusses the design and fabrication of two large-scale eFlash-based neuromorphic networks based on 55 nm eFlash memories and a general-purpose neuromorphic network fabricated in 65 nm CMOS, including $>25\times10^6$ analog 1T1R devices.

c) Hardware Security Primitives

Physical unclonable functions (PUFs) are crypto primitives that leverage the intrinsic (nanoscopic physical) variations of a system to generate unclonable secrets. Hence, PUF circuits generate a unique response even if they are similar by design and layout. They are typically classified as weak and strong based on security performance [72,73]. The former is used to generate and store secret keys and feature a relatively small access-restricted CRP space. For a weak PUF, a complete mapping function can be deduced by observing a limited number of CRPs. Strong PUFs, on the other hand, are used for authentication applications, and a complex mapping behavior is constructed by incorporating many nonlinear random components. Strong PUFs have a large CRP space and are unpredictable and resilient toward modeling attacks.

When it comes to designing PUFs, NVMs offer various advantages over conventional CMOS devices, not just due to their superior scalability prospects but because of reconfigurability and low-cost local computing capability. Besides, stochastic switching in NVMs is a rich entropy source. Such properties have been recently leveraged in building novel promising security primitive circuits. Reconfigurability could be useful when the end-user needs a new key. This is when the original one has been revealed, or the ownership is revoked or updated. Analog-grade NVMs offer potentially better security prospects considering local

computing capabilities such as nonlinear characteristics and multi-bit storage capacities. On the other hand, retaining the stored information during power shut down, NVMs may pose additional challenges to data protection [74]. In Ref. [73], we have reviewed experimental demonstrations of NVM-based security primitives. Let us mention few seminal papers in this area.

The security primitive proposed in [75] is the first demonstration that takes advantage of variations in the nonlinear *I-V* characteristics of passive memristors for building PUF. This feature is enabled by the capability of analog tuning of device conductance to maximize the security of the PUF. A CMOS-compatible 3D stack of monolithically integrated 10×10 TiO_2 memristors was used as a proof of concept to demonstrate this potential of memristors in cryptography. This work has been extended in [76] and introduced as RX-PUF (Resistive-XOR PUF). In the configuration phase, the 20×20 crossbar array with passively integrated 250 nm half-pitch devices has been programmed using a Gaussian distribution of conductances. It is showed that the devices in the tails of the distribution could weakly bias the responses, and a simple circuit technique is proposed to tackle this issue. The preliminary results showed high resiliency against machine learning attacks.

X-point PUF [77] is another design that begins with resetting all devices in a cross and then applying a single pulse to all the devices to program them to random states. The resultant conductance distribution is widely distributed between 10^{-10} to 10^{-5} S. Hence, the highly conductive devices, even if not unselected, would bias and determine the responses. X-point PUF achieves ~46% average Hamming distance on a 12×12 and has demonstrated good aging characteristics. Note that when the response is determined by a few devices, the responses remain unchanged as long as they retain their states.

Chapter 4 discusses our findings and experimental results of designing security primitives with NVMs. We show how the analog tuning and nonlinear conductance variations of memristors can be used to build a fundamental building block for implementing physically unclonable functions that are resilient, dense, fast, and energy-efficient. Using two vertically integrated 10×10 metal-oxide memristive crossbar circuits, we experimentally demonstrate a security primitive that offers a near-ideal functional performance. We present a novel architecture, called VRPUF, and prototype it using unformed 4K-ReRAM passive crossbar circuits. The architecture utilizes intrinsic process variations in crossbar circuits, manifested as variations in device *I-V* nonlinearities and the leakage currents, and allows for a huge ($\sim 10^{25}$) number of challenge-response pairs (CRPs). The VRPUF design does not require forming/ programming crosspoint devices, which simplifies peripheral circuits, leading to ~ 4× better density compared to the architectures which rely on switching the states of ReRAM devices. Moreover, uniform *IV*s of the virgin-state devices, coupled with lower conductance and stronger static nonlinearity, allow for ~100× improvement in power consumption and more robust security metrics. To boost the PUF's robustness, we propose a key-booking scheme, which dramatically improves reliability across a wide temperature range of operation and further increases PUF circuit density by reducing error-correcting overheads. Finally, introduce ChipSecure, a PUF architecture based on eFlash memories that exploit randomness in static *I-V* characteristics and reconfigurability of embedded flash memories to design efficient physically unclonable function.

2. Analog Computing with Memory Devices

Analog-grade nonvolatile memories, such as those based on floating-gate transistor [78,79], phase-change [80-82], ferroelectric [83,84], magnetic [85], solid-state electrolyte [86-89], organic [90,91], and metal-oxide [24,31,32,92,93] materials, are enabling components for mixed-signal circuits implementing vector-by-matrix multiplication, which is the most common operation in any artificial neural network. Most importantly, such circuits allow for physical-level in-memory computations in the analog domain using the fundamental Ohm and Kirchhoff laws, thus enabling dramatically higher energy and area efficiency in comparison with digital solutions.

This chapter first studies and qualitatively compares various topologies and device candidates for implementing dot-product operation in the mixed-signal domain. A brief review of major analog synaptic device candidates, i.e., 0T1R and 1T1R memristors and eFlash memories, is provided, and VMM topologies such as time-based, current-mode, and switch capacitor multiplier are analyzed from the perspectives of precision, speed, and energy efficiency.

Then, we report our results on the development of large-scale passively-integrated memristive crossbars. We demonstrate a 64×64 passive crossbar circuit with ~ 99% functional nonvolatile metal-oxide memristors featuring etch-down patterning and low-temperature budget (hence conducive to vertical monolithic integration) based on a foundry-compatible fabrication process. Most importantly, the achieved device uniformity, < 26% coefficient of variance in memristor switching voltages, is sufficient for programming a 4K-pixel gray-scale pattern with a smaller than 4% relative tuning error on average. Analog properties are also successfully verified via experimental demonstration of a 64×10 vector-by-matrix

multiplication with an average 1% relative conductance import accuracy to model the MNIST image classification by ex-situ trained single-layer perceptron. Finally, an advanced conductance tuning algorithm is proposed to reduce tuning error further, and its effectiveness is validated by simulating the performance of a multilayer perceptron classifier.

The third section expands upon the role of uniformity in passive memristive circuits and investigates how uniformity impacts the computing accuracy of analog memristive circuits, focusing on neuromorphic applications. Specifically, we explore the tradeoffs between computing accuracy, crossbar size, switching threshold variations, and target precision. All-embracing simulations of matrix multipliers and deep neural networks on CIFAR-10 and ImageNet datasets are performed to evaluate the role of uniformity on the accuracy of computing systems. Further, we study three post-fabrication methods that increase the accuracy of nonuniform 0T1R neuromorphic circuits: hardware-aware training, improved tuning algorithm, and switching threshold modification. The application of these techniques allows us to implement advanced deep neural networks with almost no accuracy drop, using current state-of-the-art analog 0T1R technology. Finally, we have analyzed the density prospects of memristive circuits and showed >5× superior density of 0T1R against 1T1R circuits, subjected to increase by downscaling the device properties.

We investigate the impact of device nonlinearity and interconnect parasitics in analog current-mode memristive VMMs in section 2.4. We show that there is an optimal tuning voltage to minimize the computation error. Furthermore, error balancing and bootstrapping are introduced as two techniques for improving computing accuracy. It is also shown that when the crossbar size is scaled up, the optimum interconnect wire conductance should increase quadratically to preserve the computing precision when using naive error balancing

approach and that the differential scheme is imperative for temperature insensitive operation and also to reduce the IR-drop effect.

Section 2.5 proposes an energy-efficient compact sensing circuit in a 55 nm CMOS process that enables a dramatic reduction of sensing circuit overhead in mixed-signal VMMs. Specifically, we argue that the redesigned eFlash memory technology is currently the most energy-efficient candidate for implementing analog synapses due to its excellent retention characteristics at deep subthreshold <100 nA regimes. Using this local sensing circuit, we develop an energy-efficient, fully current-mode analog VMM topology with digital interfaces. In this design, we use a current-mode algorithmic analog-to-digital converter (ADC), in which the analog eFlash memory is additionally used for unit current generations and offset compensation. Such considerations allow us to achieve POp/J energy efficiency at the VMM level

The remainder of this chapter focuses on the design of large-scale integrated neuromorphic computing systems and their efficient-enabling analog building blocks. A network-specific 6-layer DNN accelerator with ~1.5×10^6 training parameters and ~3×10^3 neurons is implemented. This network is fabricated with embedded Flash memories in 55 nm CMOS and could be used for performing high-speed image classification tasks, e.g., with CIFAR-10, CIFAR-100, and MNIST datasets. The second neuromorphic system is a 3-layer multi-layer perceptron suitable for the classification of 4-bit 56×56 images. Preliminary simulation results indicate the chip will be able to perform the MNIST classification task with ~98% accuracy with 100 ns/pattern and 0.25 nJ/pattern throughput and speed, far better than our previous work with 180 nm embedded Flash. Finally, we present the design and fabrication of a general-purpose DNN accelerator in 65 nm CMOS based on 1T1R memories. We discuss the

unique features of this design along with the innovative design of peripheral circuits and analog blocks which enable a compact footprint of the network, which includes ~26×10^6 1T1R cells. The chip could be used to accelerate a wide range of deep neural network inference models, e.g., ResNet-18. We also propose a lightweight, microampere-range constant with temperature (CWT) current reference generator suitable for neuromorphic accelerators. The circuit exploits a beta-multiplier block to generate a proportional-to-absolute temperature (PTAT) current. The PTAT reference is used in a block that consists of only two multi-threshold PMOS transistors and a resistor for generating a temperature-insensitive current reference. We show that when a low-threshold device (LVT) is gate-coupled with a high-threshold MOSFET (HVT), the HVT can be sized such that current variations in the LVT device due to the temperature fluctuations are minimized. A resistor is also added to the circuit to optimize the performance with respect to process variations. This simple circuit topology allows resiliency toward supply variations as well and leads to a very compact structure suitable for a distributed usage in mixed-signal neurocomputing systems. Measurement results from 4 samples of fabricated chips in a 65 nm CMOS process show an average temperature coefficient of 276 ppm/°C. The proposed circuit also achieves an average line regulation of 1.9 %/V and consumes ~15.8 μW while occupying only 640 μm^2. The last section of the chapter discusses the architecture of a general-purpose experimental setup suitable for the characterization and testing of mixed-signal neuromorphic networks.

2.1. Introduction

The rapidly growing range of applications of machine learning algorithms for image classification, speech recognition, and natural language processing has led to an urgent need for specialized neuromorphic hardware. There is much more demand for fast, low-precision

inference accelerators than for higher-precision systems for network training [94]. Despite the fact the vast majority of demonstrated accelerators from industry [71,95] and academia [65,96] belong to the category of custom digital integrated circuits, the most natural approach is based on analog and mixed-signal circuits [17,40,97,98]. Indeed, analog computing principles were developed almost four decades ago [3,7]. But, its efficient implementations are enabled only recently by the appearance of novel continuous-state, nonvolatile memory devices- the most crucial elements of analog circuits. Recent advances in analog-grade dense nonvolatile memories now enable extremely fast, compact, and energy-efficient analog and mixed-signal circuits. Such circuits are perfectly suited, in particular, for hardware implementations of the inference operation in advanced neuromorphic networks, which requires a massive amount of low-to-medium precision dot-product operations.

Floating-gate technology (or so-called "synaptic transistors") is one of the earliest forms of realizing an analog synapse, and the idea of using floating-gate transistors to implement programmable analog VMMs dates back to decades ago [7,99]. Such devices can be fabricated in standard CMOS processes, and few learning systems [78,100] have been developed based on them. One important downside of synaptic transistors is their relatively large areas (>1000F^2, where F is the minimum feature size), leading to higher interconnect capacitances, larger dynamic energy losses, and delays in massive circuits. A dramatic change in the course of this technology has been made recently by redesigning the arrays of the ubiquitous NOR flash memories with their highly optimized cells [16,101], which have enabled superior density and performance. The modification of the cell writing triples the original cell size by ×3, meanwhile allowing the cells to be fine-tuned with <1% accuracy. The density of the modified arrays (in 180 nm and 55 nm processes) is ~120 $F^{2,}$ and the technology is

commercially available to 28 nm. Another subtle important improvement stems from the fact that these cells are highly optimized in terms of retention and endurance properties.

An ideal synaptic device offers both long-term (and repeatable) analog storage and low-energy local computing capabilities in a dense structure. Analog-grade passively-integrated (0T1R) memristor is perhaps the most prospective candidate, offering excellent scalability, density, and analog memory functionality [24]. Memristor is a nanodevice with adjustable conductance G—essentially an analog nonvolatile memory cell—used at each crosspoint of a crossbar array, which (naturally and efficiently) implements the dot-product operation in the analog domain. The most appealing aspect of memristor technology is its scalability prospects. The conductance modulation in filamentary metal-oxide memristors is attributed to the reversible modulation of the concentration of oxygen vacancies. The atomic-scale of the vacancy position modulation implies the feasibility of downscaling memristors to sub-deca nanometers [24,26]. The density of a device could be as small as $4F^2$, bounded by the half-pitch metal size F. Emulating these adjustable devices with purely CMOS circuits requires orders of magnitude larger footprint. However, further progress in fabricating large memristive crossbars faces few challenges. One critical challenge is the presence of large device-to-device variations [102]. The stochastic nature of oxide rupture in such small scales complicates the reproducibility of device parameters, e.g., the voltage required for electroforming and switching. Indeed, such variabilities are the very reason for the limited demonstrations of memristive neuromorphic networks so far. One solution to alleviate this issue is the usage of selector transistors (1T1R memories) [81,98], which is inconsistent with the main driving force of this technology (i.e., scalability and three-dimensional integration compatibility).

Fig. 1 shows typical mixed-signal circuits for implementing the vector-by-matrix multiplication (VMM), the most important operation in inference accelerators and other neuromorphic tasks, while Fig. 2 provides their qualitative comparison. Specifically, due to their superior integration density, VMMs based on passive crossbars with resistive switching devices (Fig. 1.I), including metal-oxide memristors, conductive-bridge, and phase-change memories, might be the most promising in the long term. Though the integration density of the floating-gate (FG) circuits (Figs. 1.II and 1.III) is comparable with that of systems using 1T1R cells, the fabrication technology available for the latter approach is more scalable. The main relative advantage of the former approach is the FG cell's amplification, that relaxes the requirement for the gain of sensing circuitry, and enables very compact peripheral circuits. Finally, the lack of tunable capacitance devices in the switch capacitor approach (Fig. 1.IV) typically allows only 'near memory computing (instead of 'in-memory computing possible with other candidates) and leads to inferior density and other metrics.

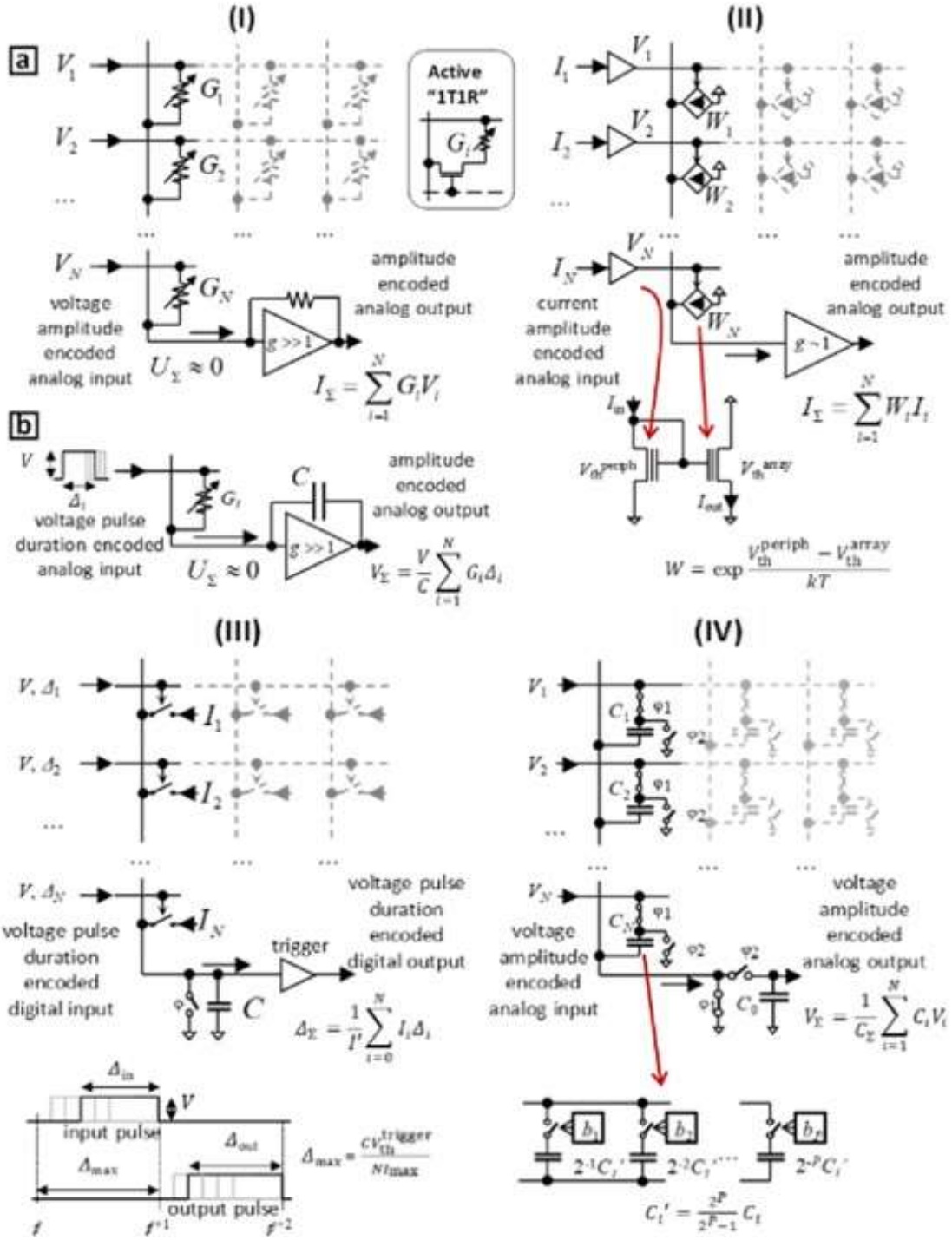


Fig. 1: Major types of mixed-signal VMM circuits: In (I), the matrix elements ('synaptic weights') are represented by continuous states of adjustable nonvolatile resistive devices (e.g., memristors), while the input signals are encoded with either (a) amplitudes, or (b) durations of voltage pulses. The top right inset shows an active ('1T1R') cell, which may also be used in circuits (a, b). In (II, III), each weight is stored in subthreshold-mode floating-gate (FG) cells, implemented as either (II) a current mirror pair formed by peripheral and array FG transistors or (III) a voltage-gated current source. In (III), both inputs and outputs are encoded

by the duration of pulses, generated within the corresponding time frame t, as shown at the bottom of panel III. In the switch capacitor approach (IV), P-bit weights are typically stored in binary-weighted fixed-value crosspoint capacitors, and the computation is performed by controlling the capacitor charge/discharge, using the switches φ1 and φ2.

Fig	Xpoint		Input/output	Density	Precision	Speed	Energy Efficiency	Maturity
Ia	0T	R	amp/amp	++	+	+++	++	-
Ib		R	time/amp	+++	++	++	+++	-
Ia	1T	R	amp/amp	+	++	++	+	+
Ib		R	time/amp	++	+++	+	++	+
II	FG		amp/amp	+	++	++	+	++
III	FG		time/time	++	+++	+	++	++
IV	C		amp/amp	-	-	+	++	+++

Fig. 2: Qualitative comparison of various VMM approaches outlined in Fig. 1: '+++' - the best, '-' – the worst. The score for precision is based on a combination of input, weight, and computing accuracies. The scores for density, speed, and energy efficiency (EE) reflect contributions from both the arrays and the peripheral circuits. Besides the maturity, all scores are for the expected level of each technology after it has been matured, rather than for its current state-of-the-art.

2.2. Developing Large-Scale Passive Crossbars

The superior density of passive analog-grade memristive crossbar circuits could enable storing extremely large neural network models directly on specialized neuromorphic chips, thus avoiding costly off-chip communication. However, to ensure efficient use of such circuits in neuromorphic systems, variations in current-voltage characteristics of crosspoint devices must be substantially lower than those of memory devices coupled with select transistors, which partly explains very limited demonstrations of neuromorphic prototypes using passive crossbars. Here we report a 64×64 passive crossbar circuit with ~ 99% functional nonvolatile metal-oxide memristors. The developed technology is based on a foundry-compatible fabrication process that features etch-down patterning and low-temperature budget, and hence conducive to vertical monolithic integration. The achieved device uniformity, most

importantly below 26% coefficient of variance in memristor switching voltages, is sufficient for programming a 4K-pixel gray-scale pattern with a smaller than 4% relative tuning error on average. Analog properties are also successfully verified via experimental demonstration of a 64×10 vector-by-matrix multiplication with an average 1% relative conductance import accuracy to model the MNIST image classification by ex-situ trained single-layer perceptron. Finally, an advanced conductance tuning algorithm is proposed to reduce tuning error further, and its effectiveness is validated by simulating the performance of a multilayer perceptron classifier. We believe that our results are a significant improvement in both complexity and analog properties over previously reported passive crossbar memories and an important step towards realizing human-brain-scale integrated neuromorphic systems.

There has been substantial progress in the development of 1T-1R memory arrays, in which a memory cell based on a two-terminal resistive switching element ("1R") also includes one dedicated select transistor ("1T"), and numerous demonstrations from academia and industry of using such active memories in neuromorphic computing circuits. Perhaps, the most impressive neuromorphic functionality was reported based on nonvolatile TaO_{2-x} devices integrated in 128×64 active crossbar arrays - see details of such devices in [103] and review of many experimental demonstrations based on such technology in [104]. The main weakness of that technology, however, is extremely large, of the order of 2500 μm^2, size of 1T1R cell, and high (mS-scale) device conductance, which necessitates bulky and energy-hungry peripheral circuits. Additionally, the reported excellent conductance tuning results are in part due to the use of the select transistor in 1T1R cell, which inhibits half-select disturbance - the main challenge for achieving high precision tuning in passively integrated circuits.

The progress in the most prospective, passive analog-grade ReRAM, however, has been much slower, mainly because of much stricter requirements for the uniformity of memory cells' *I-V* characteristics. For example, Xpoint memory - the most advanced commercialized technology using passively-integrated memory devices - operates in a digital mode. (Such memory is also most likely based on phase-change materials, which are less appealing for analog computing applications due to larger conductance drift over time.) A promising *I-V* uniformity results with very tight variations were reported for stand-alone devices based on organic [91] and epitaxial [87] materials. The main concern for these recently developed analog-grade memristors is the compatibility of the utilized fabrication flows with conventional semiconductor foundry processes. Ref. 105 describes 500-nm half-pitch 32×32 circuits based on $W/WO_x/Pd/Au$ devices, which were tuned with 25% precision (estimated from Fig. S3d data of [105]) to implement a sparse encoding algorithm. A similar device technology was recently used by the same group to demonstrate large-scale fully-integrated complementary metal-oxide-semiconductor (CMOS) / memristor circuits [106]. It is not clear, however, if the reported results in Ref. 106 were obtained based on reading conductances after completing the tuning process for all devices in the crossbar circuit or just a fraction of them, as it was performed by the same authors in Ref. 11. An even more serious and related concern is a lack of detailed statistics and, most importantly, data on retention because similar devices were shown to be volatile due to interfacial switching mechanisms, according to previous studies [107]. Another very recent work reported analog-grade 32×32 crossbar arrays based on passively integrated Si-alloy: Ag electrochemical devices [89]. Though a very impressive 100% device yield and highly linear state update characteristics were reported, the main weakness of that work is also poor retention of the devices. Additional concerns are whether

the yield results reported for 10×10 μm^2 footprint crosspoint devices will hold for nanoscale devices and the use of silver in the device stack, a contaminant typically avoided in CMOS foundry processes. Ref. 33 proposed a very promising concept for a three-dimensional memristive memory. Unfortunately, all presented experimental results in that paper were obtained for a rather unpractical structure based on microscale binary-switching devices with non-overlapping footprints so that the demonstrated three-dimensional integration does not improve the effective memristor density.

Table I summarizes experimental work on analog-grade 1T1R and 0T1R metal-oxide memristor crossbars. As evident from this table, the uniformity, density, and analog properties of previously reported memristive crossbar circuits are not sufficient for making practical neuromorphic hardware, especially for running large-scale neural models.

To address this need, our group has developed uniform CMOS-compatible fabrication technology for building larger, conducive for back-end-of-the-line 3D integration crossbar array circuits. The main contribution of this work is to mitigate the remaining challenges at the circuit and application levels and to show the prospects of such technology in neuromorphic computing applications. The developed circuits have ten times more devices and excellent uniformity allowing for significantly better array-scale conductance tuning precision as compared to the previous work [17] that reported the largest passive analog-grade memristive crossbar circuits with detailed characterization statistics. Moreover, the demonstrated artificial neural network is close in complexity to the state-of-the-art neuromorphic prototypes based on (> 10,000 sparser and 10× more conductive) 1T1R ReRAM devices [103].

Table. 2.1: Comparison of memristive circuits. The specific focus of the table is on the state-of-the-art nonvolatile (filamentary) analog-grade 0T1R metal-oxide devices, while only few

representative works are listed for metal-oxide 1T1R and solid-state-electrolyte 0T1R circuits. Note that the common concern for the solid-state electrolyte type devices (rows #1 to #3) and interfacial switching WO_x devices (rows #4 to #6) is poor state retention.

Cell type		Ref.	Crossbar size[0]	Yield (%)	Largest working demo[0]	Cell size[1] (μm^2)	Forming[2] current (μA)/ Voltage(V)	Endurance[3] (cycles)	Array level tuning precision	Set switching statistics μ / σ (V)	G_{max} /G_{min}[4] (μS)	Retention (@°C)	Type of integration / patterning technique / Substantial CMOS foundry integration challenges[5]
0T1R	Si/Ag	[89]	32×32	~100	32×32	~1200[8]	5000/3.7	>10M	-	2.25/ 0.1	10 / 1[9]	~hours	SA/RIE/High-T epitaxy&Ag
	SiGe-aSi/Ag	[108]	40×40	-	8×8[10]	0.01	-	-	~ 50%[11]	3.5/ -	4/0.1	-	BEOL / lift-off / Ag
	WO_x	[59]	11×3	-	11×3	-	1000/~1.8	-	-	0.85/0.05	-	-	SA / lift-off / none
		[105]	32×32	-	25×20	~9	>170 / -	-	~ 35%[12]	1.7/ -	3/1[13]	~mins @RT[14]	SA / lift-off / none
		[106]	108×54[15]	-	26×10	> 256[16]	-	-	-	-	2.4/1	~mins @RT[14]	FI-BEOL / lift-off / none
	Ta_2O_x	[109]	18×2	~100	18×2	-	250 / ~1.1	-	-	-	1500/850	-	SA / lift-off / none
		[110]	16×3	78	4×3	-	1000 / ~2	> 100k	-	1.25/0.1	1800/1300	-	SA / lift-off / none
	TiO_{2-x}	[24]	12×12	>50	10×6	0.16	200 / 1.9	>200k	-	0.9/0.17	200/6	>140h@76	SA / lift-off / none
		[17]	20×20	>95	17×20+8×11	0.25	220 /1.5	>100k	< 8%	1.0/0.18	200/6	>20h@120	SA / lift-off / none
		[26]	2×10×10	~100	2×10×10	0.49/2	100/2.5[17]	-	-	1.1/0.15	100/0.1	>25h@100	SA/ion beam milling/ none
		this work	**64×64**	**~99**	**64×64**	**0.5625**	**100 / 3.2**	**>100k**	**< 5%**	**1.2/0.13**	**100/6**	**>20h@100**	**SA /RIE/ none**
	HfO_{2-x}	[33]	3D 8×8[18]	-	~120[19]	~1000/8[20]	-	-	binary	-	1200/~300	-	SA / lift-off / none
1T1R	HfO_{2-x}	[25]	128×64	>99[21]	128×64	~2500[22]	-	-	<3.1%[23]	2 / -	900/100	10yr @RT	BEOL/lift-off / none
		[32]	128×8	-	960	-	>150 / >3	-	< 35%	-	40/5	-	BEOL/ lift-off / none
		[69]	128×16	>99	128×16	> 5	-	-	3.3 %	-	20/2	~ days @RT	BEOL /lift-off / none
		[111]	158K	-	158 K	1.69	100 / >1.8	< 1k	~ 2-bit	1 / -	10 / 0.1	-	NA / NA / none
		[112]	1K	-	448	~25	-	-	~ 20%	3.5 / -	100/0.1	~1m @30	NA / NA / none

[0] "Crossbar size" refers to the largest-dimension fabricated integrated crossbar circuit (not necessarily fully-functional), while the "largest working demo" refers to the largest number of devices employed at once in the demo, i.e., without relying on post-processing / combining the results from separate measurements. [1] Based on the full pitch of the integrated memory cells. [2] Largest set voltages are used if statistical data are not reported. [3] The test conditions may be different. [4] Specified at 0.1V for the devices with nonlinear static *I-V* characteristics unless noted otherwise. [5] SA = Stand-alone integrated crossbar circuit, RIE = reactive ion etching, BEOL = Back end of line integrated crossbar circuit on CMOS wafer containing access transistors, FI-BEOL = BEOL with fully integrated CMOS peripheral circuits. [6] Denser single devices are reported, though most experimental results are for 25 μm^2 devices. [7] Data for the low-resistance state. Significant retention loss at high resistance levels. [8] Based on Fig. 4c. [9] From Fig. 1d. [10] 40×40 conductance map is based on combining results from separate 25 measurements of 8×8 subarrays. [11] Based on Fig. 4d. [12] Based on Fig. S3 of [105]. Not clear if the data are obtained after tuning all devices or measured immediately after programming each device. [13] Average range of conductance values observed in the crossbar. There is a significant variation between different devices. [14] Communications with the authors. [15] 126 6×8 physical subarrays utilized for a logical 108×54 array with the conductances measured after programming each subarray. [16] From Supplementary Note 10. [17] For the top crossbar, while it is 2 V / 50 μA for the bottom one. [18] Effective crossbar dimensions based on 3D-CMOL-like structure (with overlapping electrodes in one direction). [19] Total number of employed devices in one filter based on Fig. 4d. [20] Based on Fig. 2g. Though SEM images of 300-nm-scale devices are shown, all experimental results are based on microscale devices. [21] Based on Fig.1c of [103]. [22] Based on Fig. 1c of [25, 13].

2.2.1. Device and Circuit Characterization

The developed 64×64 crossbar circuit consists of Ti/Al/TiN-based top and bottom electrodes and an Al_2O_3/TiO_{2-x} switching layer (Fig. 3). The actual crossbar array dimensions are (64+2)×(64+2), with an additional line added at both sides of the circuit for the top and bottom layers to achieve better uniformity for the devices in the main array. The details of the fabrication process are provided in [113]; here, we only focus on characterization and demonstration results.

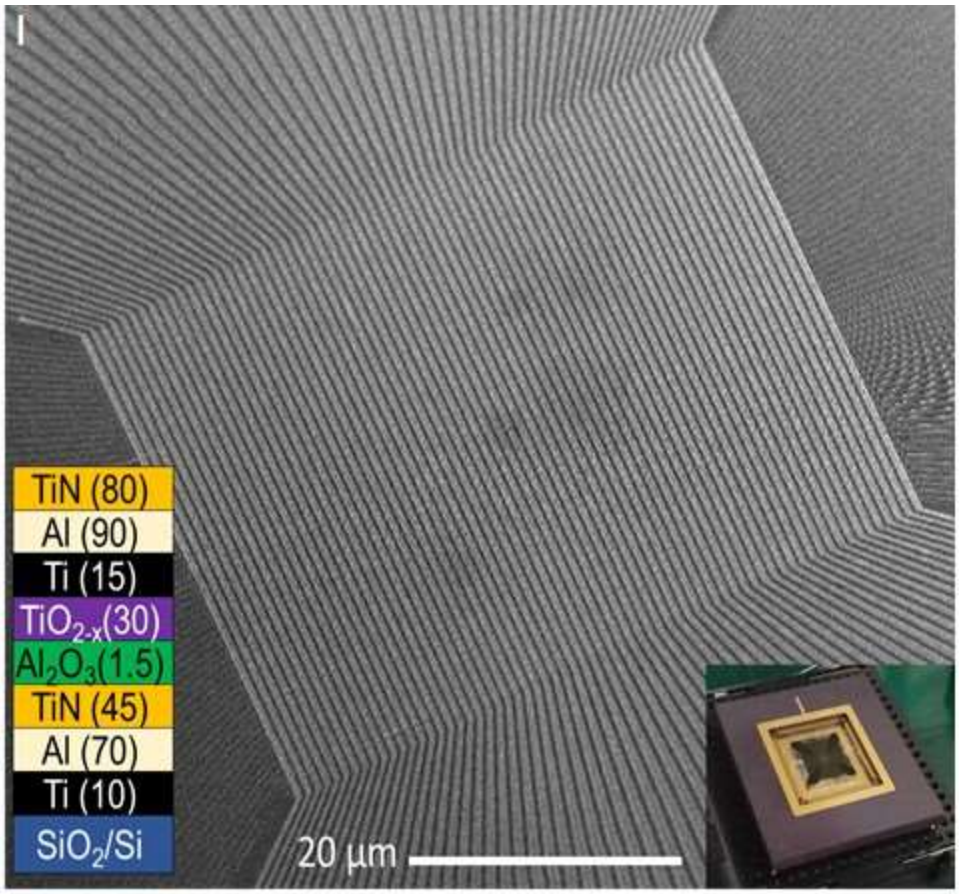


Fig. 3: SEM image of the full 64×64 memristor crossbar array. The bottom left, and bottom right insets show, correspondingly, material layers at the device cross-section with corresponding thicknesses in nanometers and the packaged chip.

Current-voltage characteristics for the as-fabricated devices, i.e., before applying the electroforming process, are fairly uniform (Fig. 4), which is an essential prerequisite for lowering variations in functional memristors [17, 24]. To electroform devices, a positive voltage is applied to the top electrode while all unselected lines in the crossbar are floated. Because of more extensive annealing compared to previous work [24], the currents via as-fabricated devices (Fig.4.4b) are just slightly less compared to the device smallest (off-state) current after forming (Fig. 5a) – see, e.g., the highlighted curves for a specific device in both figures. The forming voltages are only slightly higher on average than set voltages and completely overlap for some, making such devices effectively forming-free. The formed devices show similar in magnitude set and reset voltages (Fig. 5a), from 200 µA to 400 µA

reset and set switching currents, 2-μA-to-50-μA dynamic current range at 0.25 V, and balanced *I*-*V* characteristics, i.e., $I(V) \approx I(-V)$ at small voltages. The average nonlinearities, i.e., $0.5 \times I(V)/I(V/2)$, are ~1.1 and ~1.3 for the on and off state, respectively, at $V = 0.25$ V.

Accelerated retention tests at 100 °C are performed for 500 devices, with each device tested in 9 different random states (Fig. 5b). Post-processed experimental results and their extrapolation for room temperature operation show very promising retention characteristics (Fig. 5b,c). For example, the extrapolated results predict that the normalized conductance will drift on average by ~0.7% over 1 month at room temperature, while the average spread is expected to be less than 1.6% after 2 years (Fig. 5c). Fig. 5d confirms excellent switching endurance. It shows the results of applying 1 million tuning pulses, or effectively, switching gradually device $\sim 10^5$ times between its extreme on and off states. Note that the experiment was stopped after reaching 1M pulses because of the limitations of the experimental setup and not due to a device failure. Furthermore, decent retention was observed even after the switching endurance experiment. Fig. 5e shows measured switching dynamics characteristics for all the devices in the 64×64 array. These data are obtained by first setting the conductance of each device to 14 μS with 10% precision. Next, 1-ms-long pulses, with amplitude increased incrementally in 50 mV steps, are applied to the device. The device's conductance is read between each programming pulse at 0.25 V, and the sequence of pulses is stopped once the small-voltage conductance exceeded 50 μS. After that, we apply a similar reset/read pulse sequence until the conductance is switched back to 14 μS. The raw experimental data are used to extract effective switching thresholds, defined as the smallest amplitude of a voltage pulse at which the device conductance changes by more than 20% compared to its initial state (Fig. 5f-h). According to Fig. 5f, the average set and reset threshold voltages are 1.19 V and –1.39

V, respectively, with the standard deviations of 0.31 V and 0.37 V. Furthermore, there are only 45 (~1.125%) unswitchable devices in the whole crossbar array. The threshold maps show that faulty devices are distributed throughout the array and not contributed by faulty lines but rather stand-alone defects. These failed devices are most likely due to applying insufficiently high forming/switching voltages, which we had to bound as a precaution for avoiding permanent damage to the crossbar circuit. This, in part, is supported by the tails of the distribution in the switching threshold voltages.

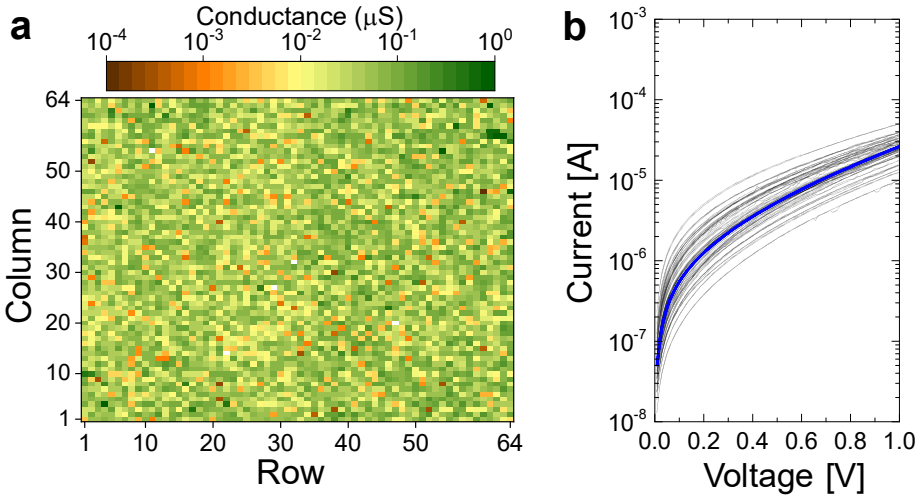


Fig. 4: As-fabricated crossbar results. (a) The conductance map measured at 0.4 V. Median conductance is ~ 45 nS. (b) *I-V* characteristics for the 36 virgin-state (i.e., before forming) devices of the 6×6 subarray located in the center of the crossbar circuit.

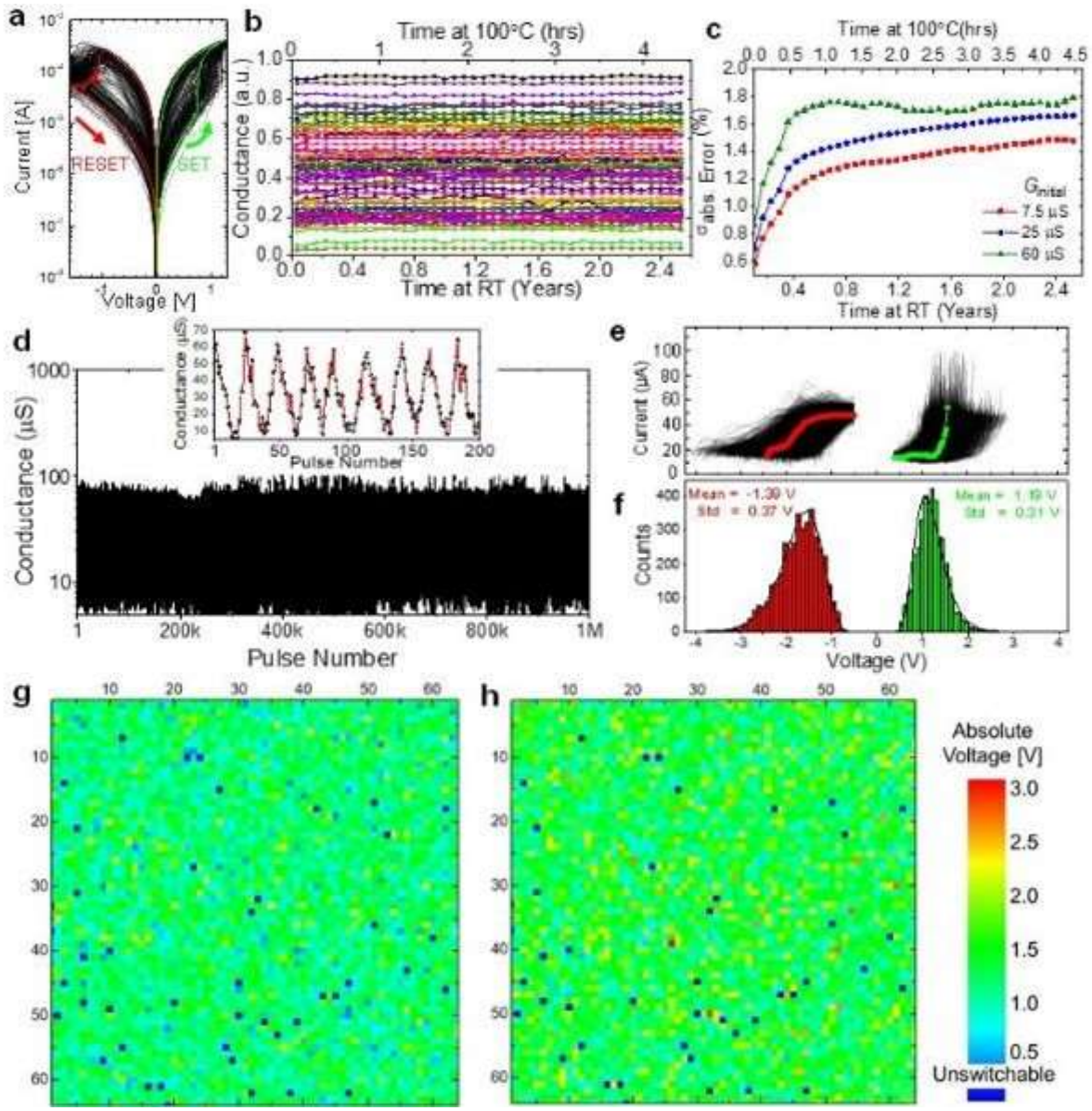


Fig. 5: (a) Representative *I-V* curves, measured with quasi-static DC voltage sweeps, for the 36 formed devices of the 6×6 subarray located in the center of the crossbar. For clarity, the curve for one particular device is highlighted. (b) Retention results for 10 different devices with data for each device shown with a specific color. The tests for each device are performed 9 times with randomly chosen initial conductance. The Evolution of the conductance measured at 400 s intervals while continuously baking the crossbar circuit at 100°C. (c) The standard deviation of the absolute conductance change normalized to G_{max} = 62.5 µS, i.e., $100\%\times|G_{intital}-G_{final}|/G_{max}$, as a function of the time interval for several ranges of initial conductances. Similar to panel b, the top axis corresponds to the measured retention data at 100°C for 500 devices, with each device tested at 6 different initial states, while the bottom axis shows extrapolated results. For panels b and c, the bottom axes show extrapolated time at room temperature assuming 1.1 eV activation energy. (d) The switching endurance results for a crossbar device. The data are obtained by repeatedly applying alternative polarity sequences of 1-ms voltage pulses. The absolute amplitude of pulse in each sequence is initially

0.8 V and then ramped up with 0.1 V steps until the device reaches the extreme (i.e., on or off) state. Inset is a zoomed-in portion of the main panel, showing typical continuous switching during the endurance test. The device is switched about 10^5 times between its extreme states during the experiment. (e) Measured evolution of conductance upon application of increasing amplitude voltage pulses. All parameters of the utilized pulse sequences are similar to those shown in Fig. 6c inset, except for 50 mV incremental step. (f-h) Extracted statistics of switching threshold voltages, defined as the smallest absolute voltage at which device conductance, measured at 0.25 V, changes by 20%, shown as (f) histogram and voltage maps for (g) set and (h) reset transitions. The conductances are measured at 0.1 V for panels b-d.

2.2.2. Conductance Tuning Experiment

The analog properties of the memristive crossbar circuits are tested by setting crosspoint device conductances using the fine-tuning algorithm [114]. Such an algorithm, which is similar to incremental step programming of flash memory devices, is based on applying a sequence of smaller-voltage non-disturbing read and larger-voltage write pulses, with a sign and amplitude of write pulses are adjusted dynamically based on the measured conductance at read pulses. An example of applying such a write-verify algorithm is illustrated in Fig. 6a, which shows the evolution of the low-voltage conductance of a specific device upon its forming, resetting to 20 μS and then tuning to 10 μS, 100 μS, and 8 μS target conductance values. Note the polarity of the tuning pulses in the inset – while applying both set and reset pulses were required because of the overshooting for tuning to 8 μS and 20 μS, only gradual resetting (setting) was sufficient to tune to 10 μS (100 μS). In fact, the device conductances can be precisely set to any value in a range from ~ 2 μS to ~ 100 μS – see, e.g., the results of device tuning with 1% relative error to linearly spaced conductance values within the lower half of the dynamic range in Fig. 6b. Fig. 6c-e shows the results of tuning conductances of all devices in the crossbar circuit, using write pulses with up to 2.5 V maximum amplitude and 4 mV / 8 mV incremental step for set/reset (Fig. 6c inset). When applying write pulses, the half-biasing scheme is adopted to reduce the disturbance of already tuned half-selected devices in

passively integrated crossbar circuits. Furthermore, to correct for a minor conductance drift in some half-selected devices upon programming, tuning of the whole crossbar is performed in several rounds, such that, e.g., all of the devices are tuned, one by one, in the first round, and then those which got disturbed beyond the specified tuning accuracy are re-tuned in the following round(s). In particular, Fig. 6d shows the map of target conductances, representing the gray-scale image of Albert Einstein mapped on all devices in the 64×64 crossbar array, while Fig. 6e shows their final values after 3 rounds of tuning. The corresponding statistics for the relative tuning error are shown in Fig. 6c. Excluding unswitchable devices, for which the error is more than 95%, ~98% of the devices are tuned within 5% relative error, while the average relative error is ~3.76%.

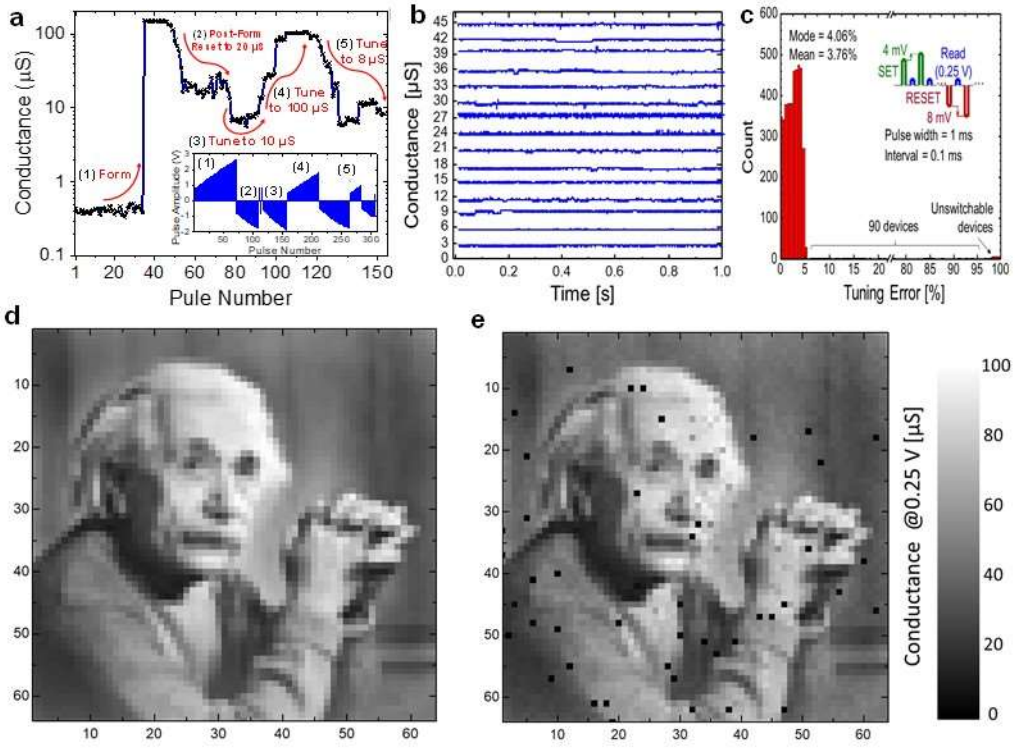


Fig. 6: Conductance tuning results. (a) Forming and high-precision tuning to 20 μS, 10 μS, 100 μS, and 8 μS target conductances of a crossbar device, with 1% relative precision. The inset shows the applied sequence of pulses during conductance tuning. Pulse sequence parameters are similar to those of panel c, except for the utilized 50 mV incremental step. (b) Example of device tuning with a 1% relative error to different conductance levels equally

spaced from 3 μS to 45 μS. (c-e) Programming Einstein image to the 64×64 crossbar array with a 5% relative error. (c) Tuning statistics. Inset shows details of the write-verify pulse sequence. (d) The target device conductances in the range of 10 μS to 100 μS corresponding to the gray-scale quantized image and (e) their actual measured values after completing automated tuning. The relative tuning error is defined as $100\times|I_{target}(0.25V)-I_{actual}(0.25V)|/I_{target}(0.25V)$. All conductances are specified at 0.25 V.

In the previous experiment, the tuning algorithm is stopped once the desired 5% relative tuning error is reached. Setting conductances with even higher precision is already demonstrated for tuning a specific device in the crossbar in Fig. 5b. The possibility of achieving higher tuning precision at the circuit level is indirectly indicated by the shape of the tuning error histogram in Fig. 6c and further verified by implementing an ex-situ trained image classifier and testing it on the common MNIST handwritten digit benchmark [40] (Fig. 7). In this experiment, we focus on demonstrating vector-by-matrix multiplication, the core operation in any neural network, while the functionality of neurons, including its bias, is emulated in the software. For simplicity, the studied network is a single-layer perceptron with 64 inputs, 10 outputs, and 640 weights. Furthermore, the original binary 28×28 MNIST images are down-sampled to 8×8 patterns so that they can be represented with 64-bit binary vectors in which black/white pixels are encoded by 0 V / 0.25 voltages and are applied to the vertical crossbar lines. Each weight is implemented with one memristor using 10 μS to 110 μS range of conductances by shifting the range of the weights upon mapping and adding pattern-dependent neuron bias at the post-processing stage. By encoding network weights with the corresponding memristor conductances G in the 64×10 portion of the crossbar, the currents measured at the virtually grounded horizontal lines of the crossbar represent the results of vector-by-matrix multiplication operation, while the output with the largest current identifies the computed class of the input pattern (Fig. 7a).

The measured classifier fidelity is compared to the software-based performance of the same network across a 1% to 50% range of weight import errors (Fig. 7d). The results show that the experimental data match simulation results closely. For example, the measured classifier accuracy for the most accurate weight import is 1.87% lower than that of the ideal software model, while the average and standard deviation for the neuron pre-activation errors are 0.61% and 0.37% - see additional details. Note that the goal of this experiment is to demonstrate the conductance tuning capabilities rather than on demonstrating high classification accuracy, which is quite low compared to the state-of-the-art numbers because of the utilized single-layer network and down-sampled B/W images. It is worth mentioning, however, that the high accuracy MNIST benchmark results for the mixed-signal circuit ReRAM-based implementations are typically obtained via hybrid demonstration in which some (small) part or functionality of a much larger neural network is experimentally demonstrated, while the rest is modeled (Table 1). Fig. 7e, f provides more details on the measured data for the two representative MNIST patterns. Specifically, the first examples show the results of the correct classification of pattern "7", with the largest current measured at the 7^{th} row of the crossbar (Fig. 7e). On the other hand, pattern "9" in the second example is misclassified (Fig. 7f). This is in part because of a large tuning error at unswitchable memristors - see stuck at high-resistance state devices at (9, 22) and (9, 24) locations in the crossbar in Fig. 7c (and also Fig. 5g, h). It is also due to narrow current margins between the correct class and the two closest classes representing digits "0" and "8", which is natural given that correct classification, in this case, would be hard even for a human.

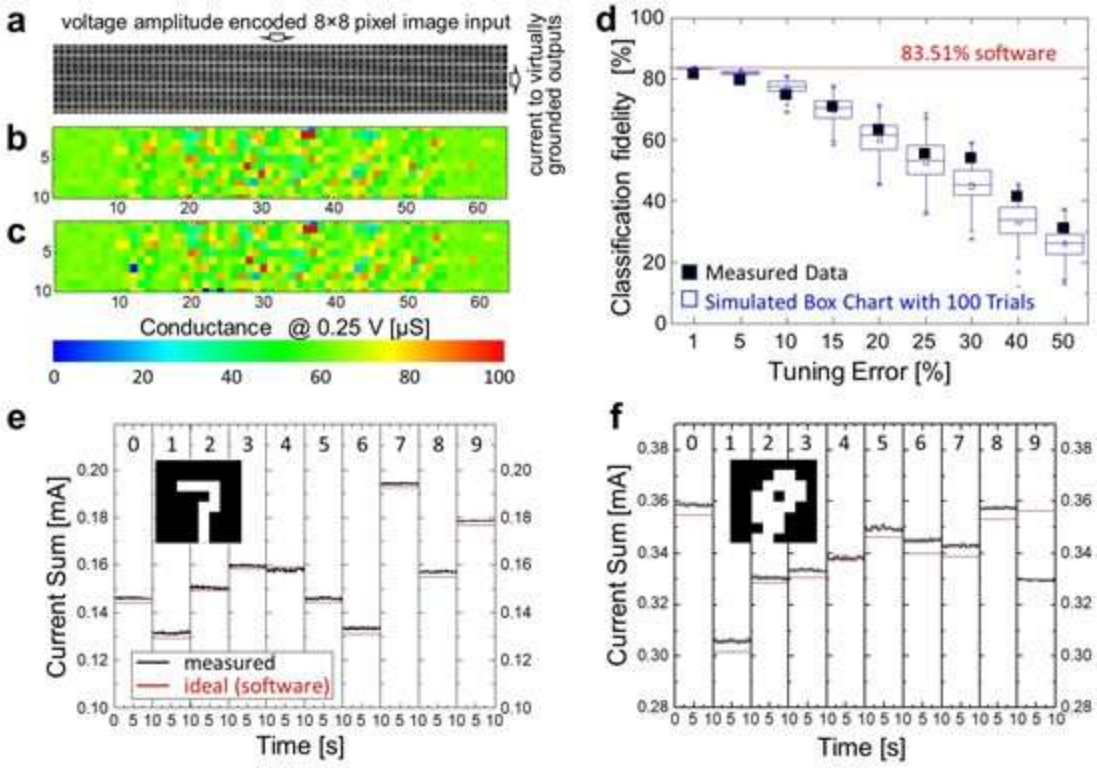


Fig. 7: Experimental results for pattern classification. (a) A portion of the crossbar circuit that is utilized in a 64×10 single-layer perceptron MNIST image classification experiment. (b) Examples of target and (c) actual conductances after tuning with a 1% relative error. (d) Measured classification fidelity and its comparison with simulation results as a function of weight import accuracy. In each simulation trial, the weights are selected randomly from a range of target_value × [1 – tuning_error, 1 + tuning_error]. (e, f) Measured output currents for all ten outputs over the 10-second interval for patterns '7' and '9' (shown in the corresponding insets) for the experiment with a 1% relative tuning error. The currents are measured, one row at a time, by simultaneously applying input voltages on all 64 columns and grounding 10 specified rows.

2.2.3. Modeling Advanced Systems

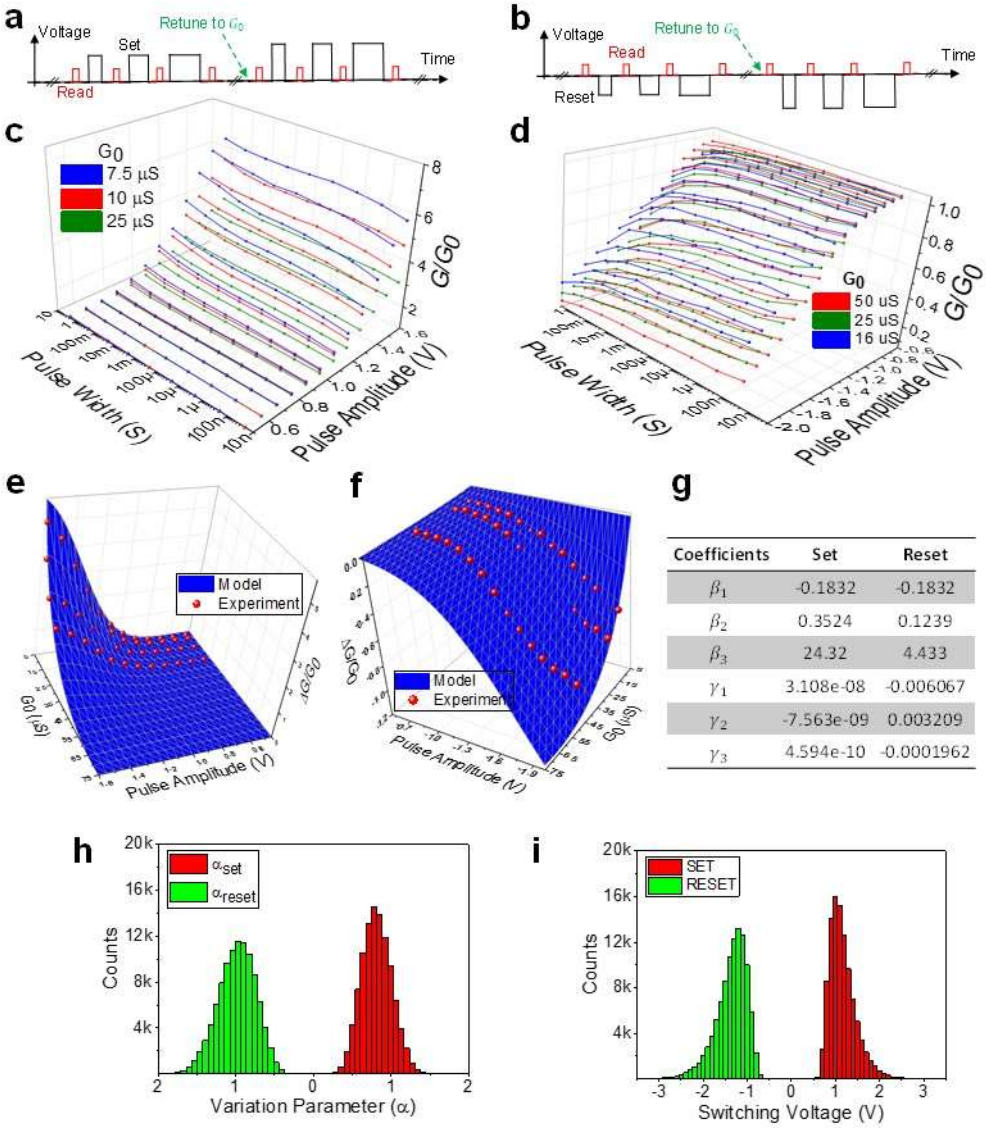
We next investigate the prospects for tuning algorithm improvements in ex-situ-trained neuromorphic inference accelerators. To make this study more informative, we develop the model for the conductance tuning process and then investigate the impact of device variations on the circuit functionality. A specific focus is on modeling half-select disturbance, which is a major challenge for accurate conductance tuning, as confirmed by experimental work. Similar to the previous work [115], the phenomenological dynamic model capturing device-

to-device variations is derived by fitting experimentally observed conductance changes for 500 crossbar-integrated memristors upon applying write voltage pulses with variable amplitude. The main purpose of the model is to estimate the change in device conductance ΔG, with respect to the initial conductance G_0, all measured at small non-disturbing (read) voltage 0.1 V, upon application of write voltage pulse with amplitude V and a fixed duration of 20 ms. The fixed duration is assumed for simplicity, i.e., to avoid explicit dependence of conductance change on pulse duration in the model. This simplifaction is also justified because of a similar fixed-duration pulse approach utilized in the tuning algorithms. Because of the long memory state retention for the developed metal-oxide memristors, i.e., their strongly nonlinear switching kinetics, obtaining meaningful experimental data for fitting conductance changes at half of the nominal write voltages required applying very long, with up to 2 ms duration pulses (Fig. 8). This is the main difference compared to the phenomenological model presented in Ref. 115, which used experimental data for a narrower range of write voltage pulse amplitudes and durations to derive dynamic model, and hence somewhat inaccurate in predicting conductance changes at smaller, half-bias voltages. The following function is found to fit well experimental data for both set and reset switching

$$\frac{\Delta G}{G_0} \approx \exp\left[\frac{\beta_1}{1+\beta_2(\alpha V)^2}\right] \sinh\left[\beta_3 \frac{\alpha V}{1+\beta_2(\alpha V)^2}\right] \left(\gamma_1 + \gamma_2\sqrt{G_0} + \gamma_3 G_0\right),$$

where $\beta_1, \beta_2, \beta_3, \gamma_1, \gamma_2$ and γ_3 are fitting parameters common for all devices (Fig. 8g), while α is a unique scaling parameter for each device that represents device-to-device variations in the switching threshold (Fig. 8h). Specifically, the model for the average behavior, with fixed $\alpha = 1$, is first found by fitting a surface to the experimental data for the average conductance changes, i.e. $\{<\Delta G/G_0>, G_0, V\}$ data points (Fig. 8e,f).

As a reminder, the effective set (reset) switching threshold of the crossbar array is defined as a voltage at which the small-voltage conductance is changed from its extreme value G_0 = 14 μS (75 μS) by more than 20%, i.e., $|\Delta G|/G_0 = 0.2$ when applying increasing amplitude positive (negative) voltage ramp. According to the fitted model, $V_{set}^* = 1$ V and $V_{reset}^* = -1.4$ V for $\alpha = 1$. The experimentally measured threshold voltages (Fig. 5f) are well approximated with log-normal distributions with parameters $\mu = 0.14$ and $\sigma = 0.25$, and $\mu = 0.29$ and $\sigma = 0.26$ for set and reset switching, respectively. According to the selected fitting function, parameter α is a multiplicative factor for the applied voltages. Hence, when modeling the set threshold voltages, we first randomly initialize V_{set} for each crosspoint device by sampling it from the fitted set threshold log-normal distribution and then find the corresponding $\alpha_{set} = V_{set}^*/ V_{set}$. A similar approach is used to initialize α_{reset}. An example of the generated α using this approach and corresponding threshold voltages predicted by the model are shown in Fig. 8h-i, respectively. Finally, since the experimentally observed variations in set and reset threshold voltages (i.e., the relative standard deviations or the coefficient of variations) are very similar, for simplicity, we use the same α when sweeping variations in the modeling studies.



Coefficients	Set	Reset
β_1	-0.1832	-0.1832
β_2	0.3524	0.1239
β_3	24.32	4.433
γ_1	3.108e-08	-0.006067
γ_2	-7.563e-09	0.003209
γ_3	4.594e-10	-0.0001962

Fig. 8: Modeling half-select disturbance. (a-d) Details of the utilized measurement protocol for modeling (a) set and (b) reset transitions and (c, d) results for conductance changes for three studied cases of initial conductance G_0. Each line with connected dots corresponds to the evolution of the conductance change, normalized to the specific tuned initial value G_0, and averaged across 500 devices upon application of the voltage pulses with a specific amplitude and exponentially increased duration. (e-i) A phenomenological model for dynamic behavior. The results of fitting dynamic equations to the experimental (e) set and (f) reset data averaged over 500 devices and (g) the corresponding model parameters. (h) The distribution of parameter α fitted to reproduce experimentally observed device-to-device variations in Fig. 14f, and (i) predicted by the model variations in the switching threshold for 4096 devices in the modeled 64×64 crossbar circuit. All conductances are specified at 0.1 V.

Using the developed model, we simulate the classification accuracy of an ex-situ-trained

784-64-1 multilayer perceptron network implemented with a hybrid CMOS / memristor

circuit under various assumptions of device-to-device switching threshold variations. The inputs to the first layer, i.e., pixel intensities, are linearly mapped to [0 V, 0.1 V] voltage range, while the inputs to the second layer are also in the same range due to the assumed clipping of rectified linear function at the neuron side. The classifier is trained ex-situ on a gray-scale 60,000 training and 10,000 MNIST test images using the conventional backpropagation algorithm with L2 regularization, 0.0005 learning rate, and 100 batch size. The software weights (w) are converted to the corresponding pair of positive (G^+) and negative (G^-) conductances using differential mapping with 33.75 μS range and 41.25 μS bias, which is roughly in the middle of the dynamic range, i.e., $G^{\pm} = 41.25\ \mu S \pm 16.875\ \mu S \times w/w_{max}$, where w_{max} is specific to the largest absolute weight value of the layer. The weight import process is simulated by first randomly initializing conductances of all memristors according to the normal distribution with 36.25 μS average and 9 μS standard deviation. A tuning algorithm based on 5-mV-step increasing amplitude pulses, starting from 0.5 V, is then applied with a 1% desired tuning accuracy, sufficient for achieving the highest classification accuracy with no device-to-device variations. To bound the tuning time for each device, the number of times for switching the write pulse polarity (when overshooting the target conductance) is limited to 5. The 785×64 weight layer, with the additional input due to bias, is mapped to 24 64×64 and 2 17×64 mixed-signal VMM blocks using the differential pair encoding of the weights. In addition to the memristive crossbar array, each block hosts a digital-to-analog converter, local sensing based on transimpedance amplifier, and programming circuitry. Such distributed implementation is similar to a mixed-signal architecture of the aCortex [67], in that the output of the local sensing circuits are currents corresponding to the partial dot-products between the corresponding weights and inputs, while the full dot-products are computed by the neuron's

(global sensing) transimpedance amplifiers by summing partial product currents. The hidden layer neurons then compute clipped rectified linear function activation and pass the results to the second layer of the network. A similar, though simpler due to the analog nature of input signals, implementation is assumed for the second layer of the network, which consists of 2 64×10 analog VMM circuits. (Note that the VMM block dimensions are chosen to match experimental work and not necessarily optimal for the studied parameters of memory devices.)

In the first studied "baseline" approach, the devices are tuned in the sequential (raster) order, similar to the experimental work. The results for the baseline algorithm show that both the tuning and classifier accuracies are significantly degraded due to half-select disturbance when the device-to-device variations (i.e., the coefficient of variation in switching threshold) are above 14% (Fig. 9b,c, and Fig. 10a-c). The second round of tuning increases the accuracy significantly, though the improvements with additional rounds are negligible (Fig. 10a). The simulated absolute tuning error at 26% device variations is ~ 9.6% (Fig. 9b), which is higher compared to the experimental results in part because of the excluded (unswitchable) devices. Three different techniques are further proposed to improve the conductance tuning process. In the first technique, the write voltage amplitudes are bounded within a certain range of voltages, with the range gradually reduced with each round of tuning (Fig. 10). Such an approach results in better average tuning accuracy compared to the baseline approach when device variations are higher, at the cost of abandoning of tuning the devices with a larger threshold switching voltages (Fig. 9b). In the second technique, the devices with high set (> 1.75 V) and reset (< -2 V) switching voltages are first identified. The high-set threshold devices are then switched to the highest conductive (>75 μS) state used in weight mapping, while the high-reset devices are switching to the highest resistive (<7.5 μS) before the tuning

algorithm is applied. Such presetting significantly reduces the use of larger amplitude write pulses throughout the tuning process and hence minimizes half-select disturbance, especially when applied together with the first technique – see the results for approach #2, which utilizes both techniques in Fig. 9b. The third technique takes advantage of the possibility to encode the same weight with different target conductances in the differential pair implementation, i.e., by shifting the conductances of a pair by the same amount. In particular, when the maximum voltage limitation of the first technique is met, the target conductances of a pair are adjusted, and the conductance tuning of another device in a pair is attempted instead. Application of all three techniques (approach #3) significantly improves the tuning accuracy, e.g., improving it by 9% as compared to the baseline approach for the case of 26% device variations. More importantly, at such device variations, the classification accuracy of the baseline approach is significantly improved to ~97.3%, which is within 0.7% of the highest possible accuracy for the studied network, while the highest amount of device variations, which can be tolerated without losing classification accuracy is increased from ~14% to ~20% (Fig. 9c and Fig. 10c,f,h,l).

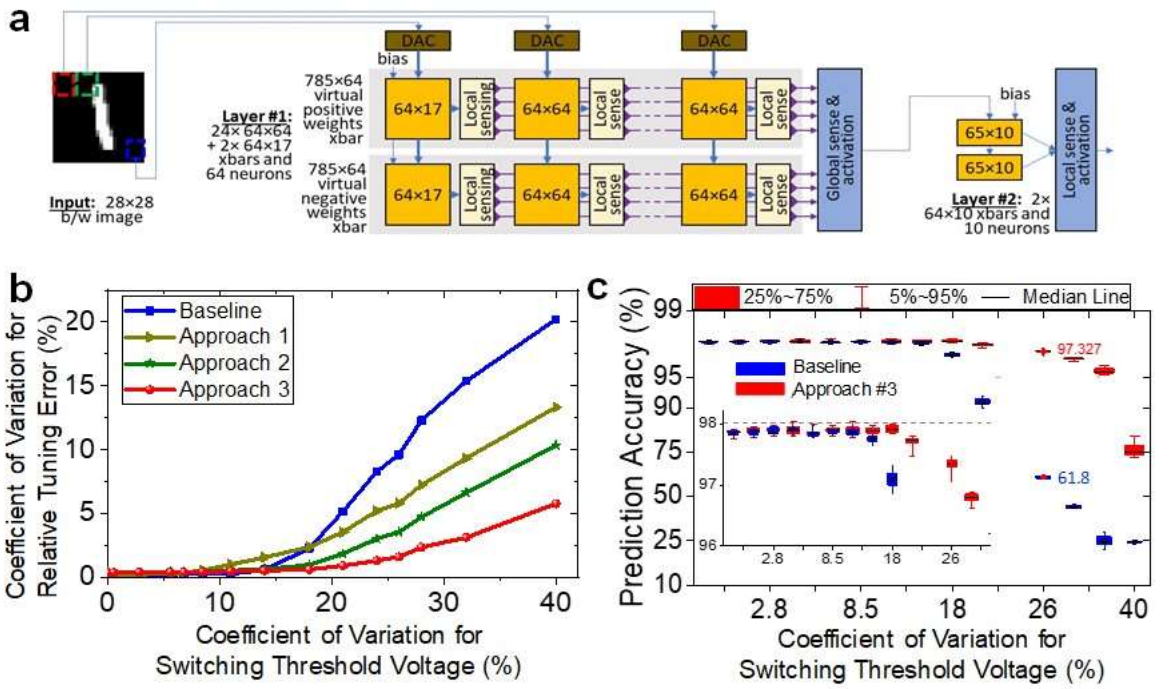


Fig. 9: Modeling of ex-situ trained MLP classifier. (a) The block diagram for the distributed mixed-signal implementation of 784-64-10 multilayer perceptron (MLP) classifier with a 64×64 crossbar circuits. Programming circuitry is omitted for clarity. (b) Modeled relative tuning error after 10 rounds of tuning for the first layer of MLP network as a function of device-to-device variations when using four different conductance tuning approaches. See Section "Modeling of Advanced Systems" for more details of the tuning approaches. (c) Simulated accuracy of MLP classifier as a function of device-to-device variations when using baseline and the most advanced tuning approach after 10 rounds of tuning. Inset shows zoom-in for the high classification accuracy portion of the graph. The shown numbers roughly correspond to the device variations observed in the experiment. The box plot shows the statistics over 10 different runs of initial conductances. For simplicity, memristors' static *I-V* nonlinearities and noise are neglected, and ideal peripheral circuits are assumed in simulations.

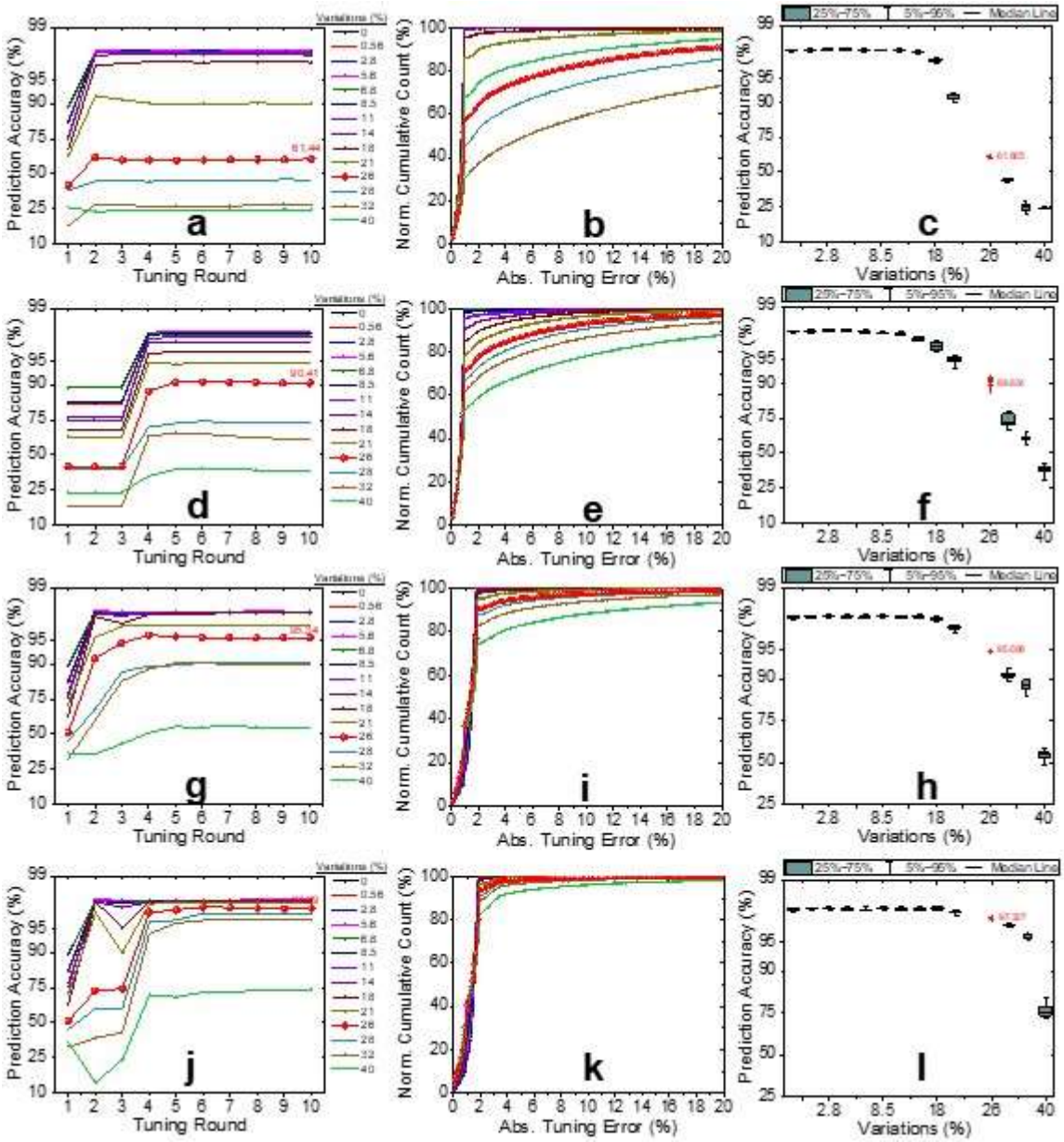


Fig. 10: The impact of device uniformity on the accuracy of MLP. (a-l) Modeling results when using (a-c) baseline, (d-f) 1st, (g-h) 2nd, and (j-l) 3rd tuning approaches. (a, d, g, j) The improvements in classification accuracy with more rounds of tuning. (b, e, i, k) Cumulative distribution of the absolute tuning error at the end of the 10th tuning round. (c, f, h, l) Classification accuracy as a function of device variations at the end of the 10th tuning round. The box plot shows the statistics over 10 different cases of initial conductances. The thick red lines correspond to the demonstrated technology, i.e., α = ~26%. For simplicity, memristors' static *I-V* nonlinearities and noise are neglected, and ideal peripheral circuits are assumed in simulations. In panel a, the accuracy saturates after a few rounds because of the significant half-select disturbance when re-tuning higher switching threshold devices. In panel d, the utilized maximum values for set / reset thresholds are NA, 2, 0, 1.65, 1.45, 1.4, 1.35, 1.3, 1.2, 1.1 / NA, 0, -1.65, -1.45, -1.35, -1.3, -1.2, -1.1 for tuning rounds #1, #2, …, #10, respectively. In panel j, the highest accuracy is 97.29%.

2.2.4. Discussion

In summary, the general goal of this work is on increasing the complexity of passively integrated memristive crossbars and developing a fully CMOS-compatible process while maintaining high yield and sufficiently low spread in current-voltage characteristics of integrated metal-oxide memristors, one of the most critical problems prohibiting practical use of this technology in neuromorphic computing applications. Our main contributions include the development of uniform 64×64 passive crossbar circuits with almost 99% working crosspoint metal-oxide memristors based on foundry-compatible fabrication process suitable for back-end-of-line / 3D integration and experimental demonstrations of conductance tuning with < 4 % relative average error for programming 4K gray-scale pattern and close to 1% error when implementing 640-weight ex-situ-trained single perceptron network. Additionally, we propose the advanced tuning algorithm and verify its effectiveness by simulating a multilayer perceptron.

2.3. The Role of Uniformity in Passive Memristive Circuits

In the previous chapter, we discussed how the stochastic nature of oxide rupture in small scales complicates the reproducibility of memristor parameters, e.g., the voltage required for electroforming and switching, and illustrated how such variabilities are the very reason for the limited demonstrations of memristive neuromorphic networks so far. We qualitatively made this case that the prospects of using 0T1R for building large-scale deep learning systems are excellent due to the superb density of this technology. We demonstrated the successful integration of 64×64 passive metal-oxide memristor crossbar circuit, which reports sufficient uniformity for <5% average tuning precision, slightly worse than ~3% reported in analog 1T1R memories [13].

However, there are some key questions regarding the role of the process uniformity in the design of passive memristive crossbars, which needs further clarification: How is the crossbar uniformity related to the computing accuracy, and what are the critical factors affecting it? How can we build more resilient circuits and improve the performance in higher (than device fabrication) design levels? How much crossbar uniformity is needed to achieve software-equivalent accuracy and build a passively-integrated memristor-based large-scale deep neural network? This section aims to expand upon these important questions and the critical role of switching threshold variations in the computing precision of neuromorphic networks. First, we discuss the preliminaries, motivation, and previously fabricated analog-grade memristor crossbars. Then, the simulation framework is illustrated. Further, extensive simulations of vector-by-matrix multipliers and representative neuromorphic networks are performed to assess the tradeoffs and trends. Finally, three post-fabrication solutions are explored for improving the performance of neuromorphic circuits. The section is concluded with a thorough discussion of the results and prospects of harnessing 0T1R and 1T1R circuits in neuromorphic circuits.

2.3.1. Preliminaries and Motivation

Memristive devices are typically operated in three phases: forming, programming, and read. Upon fabrication, devices are initially in the pristine state and require a one-time forming process before becoming adjustable memristors. The electroforming process includes applying a current-limited ramp voltage to a device and continuously monitoring its low-voltage conductance. When the device reaches a certain threshold, a conductive filament forms inside it [116], and its low-voltage conductance jumps significantly, enabling subsequent analog-state tuning and storage. In the second phase, the tuning or programming

stage, the conductance of the device is adjusted to a desirable value (G) through the modulation of the impurity profile. We may increase (set) or decrease (reset) the conductivity of the device by applying a moderately large voltage to the device that is about (or slightly larger than) its switching threshold—a device-unique voltage that alters its conductance by, e.g., 20%. Harnessing the write-verify algorithm, we keep programming and monitoring the state of the device ($\hat{G}$) until reaching a certain tuning accuracy $(\hat{G} - G)/G < \in$. Ultimately, to implement multiplication, summation, or useful computational tasks, devices are operated in the non-disturbing read (inference) phase: A relatively low voltage (V) is applied to the device, and the generated current, $I=\hat{G}.V$ is sensed in a CMOS circuity. Here, we are interested in investigating how the parameters of a 0T1R memristor technology, i.e., the variations in the switching thresholds, are related to the tuning accuracy ($\in$) and, in turn, the computational accuracy of memristive neuromorphic networks. When a high-precision readout circuit is available and memristive devices have excellent retention characteristics, $\in$ is almost entirely bounded by the dynamic switching characteristics of the devices.

To clarify this, consider the practical V/2 approach of tuning memristive crossbars (Fig. 11a). The voltage applied on the selected device (by peripheral decoders and switch matrix) is V_{set}. Unselected electrodes are pinned to $V_{\text{set}}/2$ to minimize the disturbance on other devices. The applied voltage on the unselected devices is zero; however, $V_{\text{set}}/2$ is dropped on the devices which share an electrode with the selected device (i.e., half selected devices). If the switching threshold of these devices is $\sim V_{\text{set}}/2$ or less, their state shifts undesirably, resulting in an imprecise tuning. A similar idea also holds for the reset operation. Fig. 11b shows the measured IV characteristics of a device (R_0) in the 64×64 crossbar. Two hypothetical switching threshold distributions, as well as presumptive IV characteristics of

two corresponding devices, are included to clarify our point. When we set R_0, $V_{set,R0}/2$ drops on both R_1 and R_2. The state of R_1 is expected to alter negligibly since the set threshold of R_1 is much larger than $V_{set,R0}/2$, unlike R_2 that switches considerably. Hence, when tuning the entire crossbar, the total disturbance is correlated to the variations in the distribution of switching thresholds, and the smaller variations (or higher uniformity) result in a higher tuning precision.

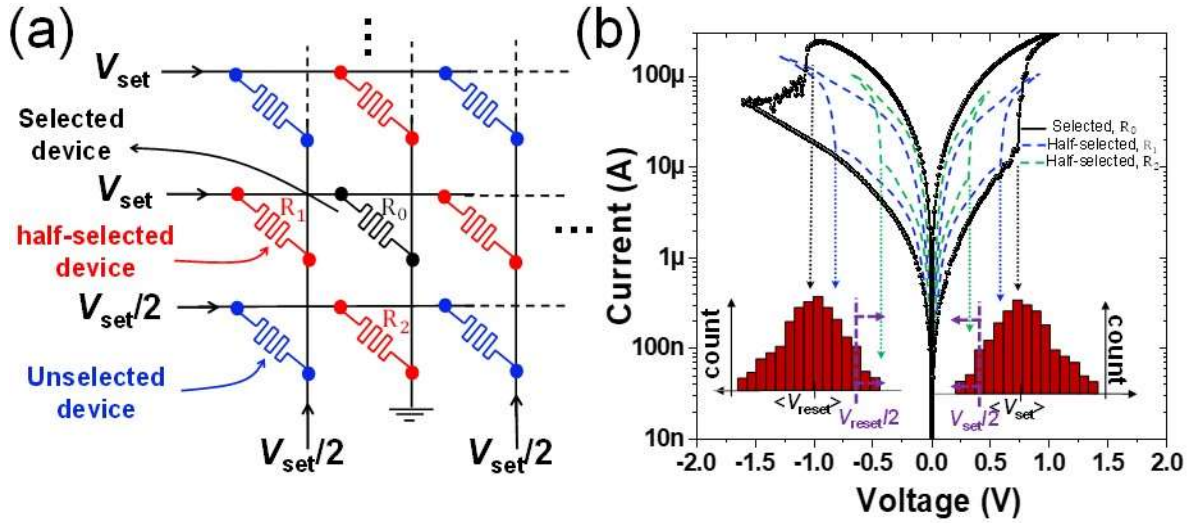


Fig. 11: (a) The schematic of the 3×3 portion of the crossbar and the V/2 tuning scheme with highlighted selected, unselected, and half-selected devices. Panel (b) shows a typical *IV* characteristic of a device and reveals why the tight distribution of switching voltage is critical.

TiO_2 memristors have been used in designing 10×2 [2], 12×12 [24], and 20×20 [17], and 64×64 [113] crossbar circuits, with excellent retention (>20 hrs in 100°C), endurance (> 10^6 analog switching cycles), and close to 100% yield. The normalized variations in these works are 10%, 11%, 18%, and ~26%, respectively. The same stack is also used in the only analog-grade 3D integrated demonstration [26] using two layers of 10×10 memristor array and reporting ~13.6% normalized variations. The consistent trends in the TiO_2 crossbars indicate that the larger the crossbar size, the higher the normalized variations. This trend partially stems from the fact that (the worst-case or the largest) forming current required for electroforming increases with the crossbar size owing to the increase in extra leakage current. Hence, a larger

compliance voltage/current is required as more and more devices are electroformed, which increases electrical stress and, ultimately, leads to a higher device variability. When forming our 64 × 64 crossbars, the maximum electroforming current is set to ~50 μA at the beginning, but it is raised to ~ 1 mA or so at the end. In addition to this observation, the more devices in the crossbar, the more disturbance created during tuning. As it will be shown in this paper, these factors amplify each other and exponentially entangle the design and operation of larger analog crossbar circuits. Fortunately, preliminary architectural studies show that for many computing applications, e.g., deep learning accelerators, the optimum crossbar dimension is in the range of 64×64 as choosing enormous crossbar modules underutilizes the hardware resources and reduces the overall performance [67].

Regardless, the relationship between the normalized variations and the crossbar size with circuit fidelity is unclear and requires further research. To clarify it, we first use tremendous experimental data to develop a reliable dynamic model of the memristor that relates the conductance change to the switching thresholds and the applied voltage. Then, we use this model to emulate the tuning process of ex-situ weight transfer and find the relationship between the accuracy, block size, and normalized variations in general VMM blocks and representative neuromorphic circuits.

2.3.2. Simulation Framework

In order to study the role of uniformity in memristive crossbar circuits, we use the dynamic, which is comprehensively discussed in the previous section (more information is also available in [117]). Let us emphasize several important points prior to discussing our results. We could not use previous models (including our recent work [115]) since the switching threshold parameter is not used in those models. Though we could consider a very

general model, we prefer to use a model that perfectly fits the behavior of an experimentally fabricated stack that meets the essential requirements for analog computing (analog tunability, high retention, endurance, etc.). We believe that this strategy improves the reliability of our results. Besides, since our candidate device and other metal-oxide memristors share some similar switching characteristics, we expect to see resembling trends of the results for other devices as well. Last but not least, we consider a fixed pulse shape and duration to avoid overcomplicating the study and improve the goodness of our modeling fits. Note that we also use square-shaped pulses (typically with fixed amplitude) in practice to reduce the impact of cycle-to-cycle variations in the runtime [9]. In the following, we discuss how we emulate the tuning and perform the ex-situ training using this dynamic model.

In ex-situ training of a neuromorphic network, synaptic weights are calculated on a precursor software-based network and then imported sequentially into the crossbar circuits. Networks are typically composed of many crossbar blocks which are programmed in parallel or sequentially. However, within a crossbar, the devices are tuned into their corresponding predetermined desired states, individually (one-by-one). Due to the stochastic nature of the switching mechanism in memristors, particularly analog-grade devices, often require multiple pulses to reach an absolute accuracy. This is executed using the well-known write-verify algorithm [102]. In every simulation case, the conductances of devices are initially randomized using a Gaussian distribution with an average of 36.25 μS (midrange conductance) and a standard deviation of 9 μS. Then, conductances are adjusted one-by-one using the write-verify algorithm and the dynamic model. We reconstruct the exact procedure that we employ in the experiments when tuning the devices. The devices within any crossbar block are tuned in raster order. More importantly, to increase the tuning speed, we

progressively increase the pulse amplitude (set/reset) starting from 0.5 V with 10 mV steps to the switching voltage of the device. This idea also prevents overstressing the device. The tuning direction (setting or resetting) is alternated whenever we pass the target conductance. To avoid overstressing the memristors, creating too much disturbance, and reducing the tuning time, we limit the tuning process for every device to 5 rounds. The algorithm is aborted (and restarted with the next device) whenever it reaches the desired tuning accuracy or the maximum permitted pulse per device. The half-select disturbance is applied for every applied pulse and every device by updating the state of devices sharing either top/bottom electrode with the V/2 rule. The entire crossbar is tuned for 10 rounds to diminish the disturbances.

2.3.3. Computing Precision in Nonuniform Crossbars

VMM is the most critical operation in inference accelerators and most neuromorphic tasks. The fidelity of most neural network models closely follows the computing precision in their VMMs. Here, we consider $N \times N$ two-quadrant VMM circuits, which are implemented in the analog domain by two separate $N \times N$ memristive crossbars. VMM size, variations in switching thresholds, and target precision are variables of this research. For every case study, 20 crossbars with random log-normally distributed switching thresholds, and 20 different normally distributed weight matrices with zero mean are generated. The mapping function $G_{ij}^{\pm} = G_{\min} + (1 \pm W_{ij})(G_{\max} - G_{\min})/2$ in which W_{ij} is the normalized weight and $G_{\max}$ and $G_{\min}$ are upper and lower conductance bounds are used to convert dimensionless weights into device conductances [22]. For each VMM, we randomly generate 1k input voltage vectors, with elements uniformly distributed in the range 0 to 0.1 V. VMM computing errors are then calculated over the output current (I) and defined by $|I_{\text{actual}} - I_{\text{ideal}}|/I_{\max}$. Ideal currents ($I_{\text{ideal}}$) are obtained directly from the mathematical vector-by-matrix multiplication

of the input voltage vector and conductance matrix, actual currents (I_{actual}) are obtained from the circuit simulation after all devices are tuned, and $I_{\max}$ is the maximum absolute pre-activation current over all input combinations.

First, the mechanism of error propagation is investigated for 64×64 VMMs and 5% and 25% variations. Fig. 12a shows the tuning error for 50 devices (in the crossbar that implements G^+) during 10 rounds of the programming phase in the case with 5% variations. Specifically, each curve shows how the tuning error for each device evolves starting from the first attempt to the last one. One curve is highlighted for better clarity. The steep drops in each curve denote the moments the device is tuned. For the highlighted curve, the device is initially tuned with <1% accuracy, but the disturbance moves its state leading to ~5% error by the end of the 1st round. The device is retuned in the 2nd round, and the disturbance alters it to ~3% of the target. Less disturbance generated in the 2nd round stems from the fact that some devices are within the target accuracy by the end of the 1st round. So, the total number of pulses (and hence overall disturbance) decreases in each round. The state of most devices stabilizes by the end of the 4th round. The conductance error distributions and related statistics, shown in Fig. 12b, confirm these findings as well. Note that the assumption of 5% variations in a 64×64 crossbar is too ideal with the current technology. Figs. 12b,c show the result from the simulations of crossbars with 25% variations in the switching thresholds. Though the result slightly improves in the first 4 rounds, many devices remain in imprecisely tuned states after that. The periodic state evolution of many devices (e.g., the highlighted curve) in Fig. 12c is because of the large disturbance and tremendous dependencies, making the tuning effectively unstable for many devices. Fig. 12e compares the ultimate distribution of conductance error for both cases. The 99 percentiles of the tuning error are ~14.4 % and ~1.0 % for 25% and 5% variations,

respectively. The huge gap between the realistic and ideal case signifies the importance of variations in passive crossbars. Imprecise tuning results in a large error in the output signal, as expected. Fig. 12f shows the pre-activation error distribution for both cases. The 99 percentiles of the distributions are ~7.0 % and ~3.7 % for σ_n=25% and σ_n=5% variations, respectively.

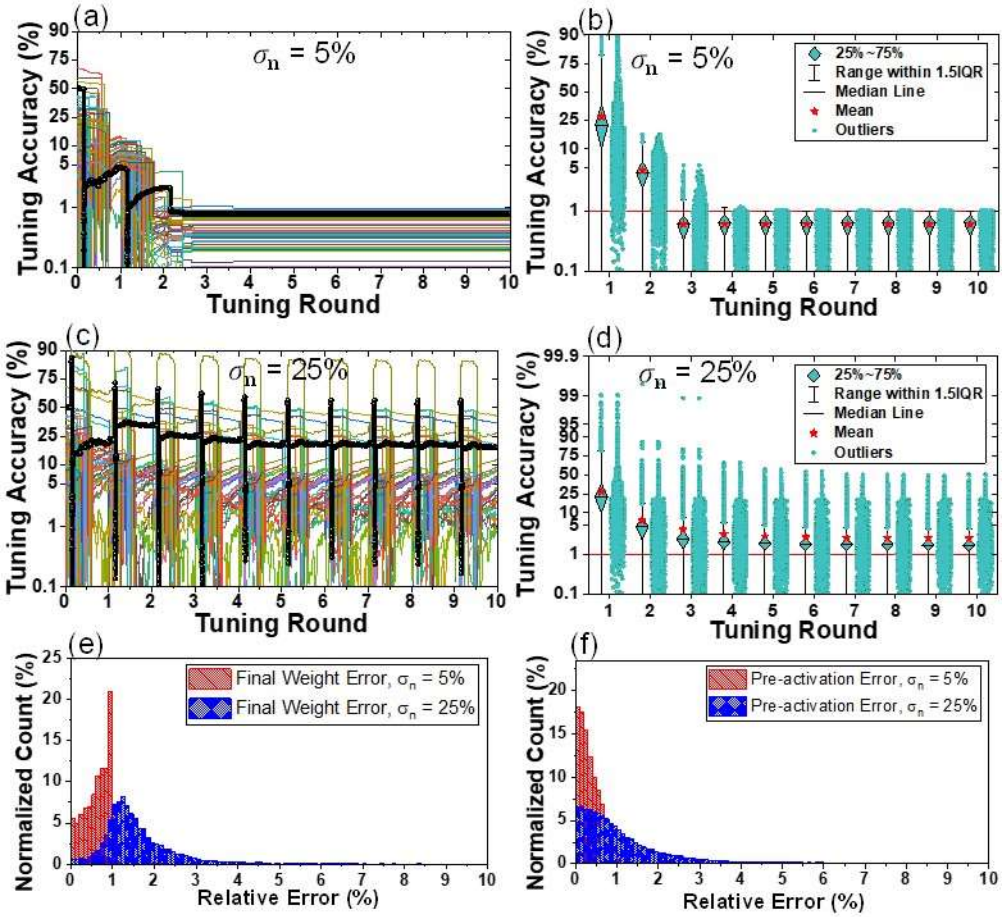


Fig. 12: Tuning analysis in a 64×64 VMM with random weights for two cases of 5% and 25% of normalized variations in switching thresholds. Panels (a,b) and (c,d) correspond to weight tuning error for 5% and 25% variations, respectively. Panels (a and c) show the evolution of tuning error for 50 devices during 10 rounds of tuning the crossbar. Panels (b and d) denote the error distribution for all devices at the end of each round. Panels (e) and (f) show the distribution of weight and pre-activation errors (at the end of the 10th round) for the two cases, respectively.

Fig. 13 summarizes our VMM-level simulation results in which the role of VMM size, switching threshold variations, and target tuning error are studied. In every data point, we

consider 400 VMM instances (20 different sets of weights and 20 crossbars) to characterize the worst-case error statistics (99 percentiles of the output error among 10^3 patterns), the de facto parameter to evaluate the computational accuracy. Note that the VMM size and normalized variations are increased exponentially and linearly, respectively. Let us emphasize that target tuning error is an implicit parameter that is determined by how precisely a single device can be tuned (disregarding the rest of the devices). The target tuning error depends on the switching characteristics and retention of the devices. In every panel of Fig. 13, the dashed red line serves as the intrinsic bound and shows the expected intrinsic error resulting from the imprecise tuning of individual devices (without the half-select problem). Such intrinsic error is often linearly proportional to the target tuning error. The first observation is that the median worst-case error increases exponentially with variations, more evidently for $N>30$. It also increases exponentially with respect to VMM size for low variations and super-exponentially in large variations. This stems from the fact that when variations become more extensive in large circuits, the crossbar moves toward instability, and an avalanche-like phenomenon (Fig. 12c) prevents the precise tuning of the majority of the devices. The spread of the worst-case VMM error distribution among different instances also extends with increasing size or variations in high precision tuning cases since the chance of hitting worse corner cases raises when disturbance escalates. This issue becomes particularly important in high-precision computing tasks with tight error margins.

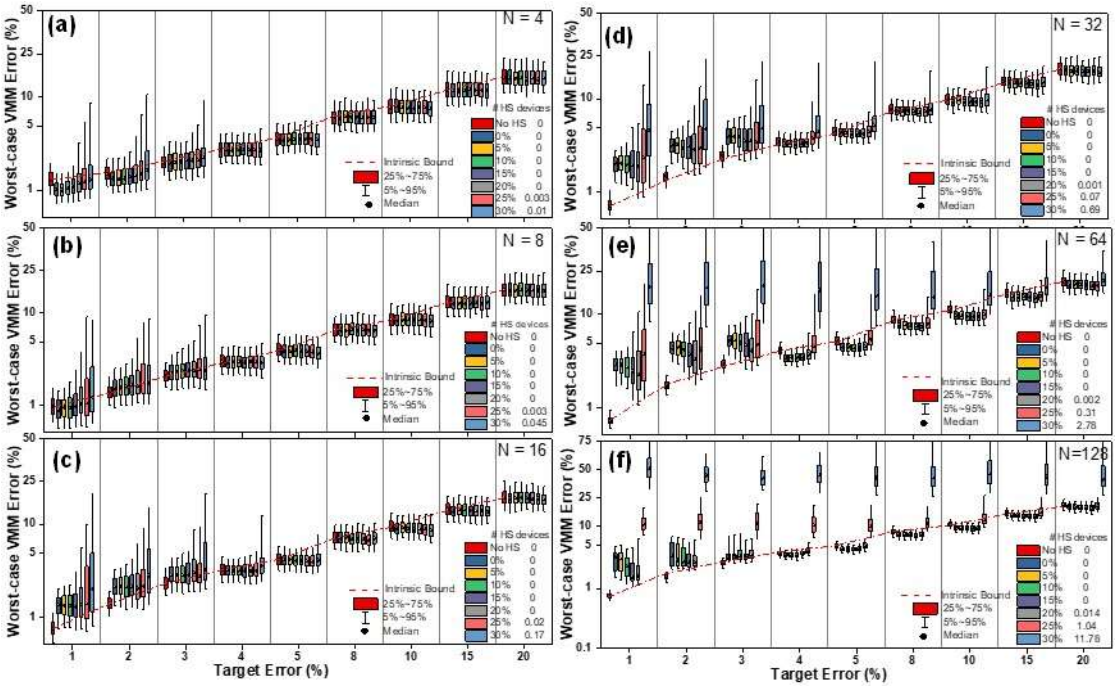


Fig. 13: The distribution of the worst-case VMM error (99 percentile of the absolute error distribution for 10^3 patterns) among 400 instances of a) 4×4, b) 8×8, c) 16×16, d) 32×32, e) 64×64 and f) 128×128 VMMs. No HS: no half-select, n_{HS}: the average number of devices affected with V/2 disturbance normalized by N^2. n_{HS} is a metric that shows the overall disturbance level in a certain crossbar. The median for the 'no half-select' case is considered the intrinsic bound and shows how accurately we can compute given a semi-exponential tuning error distribution in the crossbar. Note the log-scale of the y-axis in all panels.

For small VMMs (e.g., N<16), the error follows the intrinsic trend even in the presence of large variations because the total disturbance is low enough to be fully recovered after running the algorithm several iterations. In moderate VMM sizes (e.g., N=32), the error tends to increase for high precision tuning cases (e.g., <4%), particularly when the variations are high. This error escalation originates from an increase in the number of applied pulses for achieving a better tuning precision, which in turn leads to a larger disturbance. For large VMMs, variations become more prominent such that the computational accuracy is adversely impacted. For N=64, the drastic change for σ>0.25 also stems from the exponential growth of the overall severe half-select disturbance cases. To clarify this, let us look at n_{HS} —the normalized number of devices whose switching thresholds are equal to or less than half of

another device in the crossbar. The value of n_{HS} for each case study is provided in each panel of Fig. 13. When the standard deviation of the set threshold distribution increases from 0.25 to 0.3, n_{HS} soars by a factor of ~10, indicating a surge in the cases of severe disturbances.

Another subtle point is related to the reduced computational accuracy in cases with even no variations. For instance, comparing the case of no half-select (no HS) with σ_n=0, we observe a ~4.3% increase of the average error in the case of *N*=64 and 1% target. Even with no variations, the voltage drop on other devices could have a slightly disturbing effect (due to the limited *IV* slope), which could become potentially noticeable when the total number of pulses grows very large. Note that this issue might become overwhelmingly problematic for crossbars with poor retention as they often have a much smaller slope. The slight improvements beyond the intrinsic error (e.g., see the case of *N* = 4 and target error = 5%) originate from the regularization impact of half-select disturbance, which slightly improves the accuracy. Finally, the computational accuracy of state-of-the-art 1T1R and 0T1R crossbars can be compared in Fig. 13. For practical VMM sizes (e.g., *N*=64), considering state-of-the-art reported 1T1R (no half-select) tuning accuracy (~3% in [13] using highly conductive devices), the computational accuracy of 0T1R VMMs (σ<30) is the same as 1T1R, when tuned with 1% target precision or worse by ~1% when tuned similarly with 3%. Such comparison indicates excellent prospects of using 0T1R design in massively large neuromorphic computing networks since passive crossbars are much denser and 3D stackable and evidently offer comparable computing precision.

The final takeaway is that the computational accuracy in passive crossbars is a function of the total number of applied pulses or total disturbance from the tuning precision perspective. Note that the larger the VMM, the larger the number of pulses; the larger the variations, the

more pulses needed to tune the devices in multiple rounds; the smaller the target tuning; the higher the number of pulses. System-level and architecture considerations often determine an optimum kernel size (N) to optimize the functional performance. Hence, there are two ways to improve the computational accuracy in passively-integrated-based neuromorphic systems: fabricating more uniform crossbars that lead to tighter variations and developing more optimum tuning algorithms that directly reduce the total disturbance and the number of applied pulses. Further, the most efficient and accurate circuit is not necessarily obtained when the device is pushed to its high precision limit. Hence, extensive simulations (similar to our work) are required to find the optimum tuning margin for a given technology, kernel size, and the computing model.

2.3.4. Neuromorphic Network Benchmarks

We consider 2 representative neuromorphic networks: a moderate-size convolutional neural network (ConvNet) and ResNet-18. The former is a modified Lenet-5 architecture that includes 2 convolutional, 2 pooling, and 2 fully-connected layers (see [118] for more details on the structure). The model is trained with 50k images and tested on the remaining 10k images of the CIFAR-10 dataset. Standard data augmentation techniques such as zero-padding with two pixels, cropping a random 32×32 region, and performing random horizontal flipping of images are employed. We use ADAM optimizer, cross-entropy cost function, a batch size of 64, a learning rate of 0.001, and 220 epochs to achieve 87.25% inference accuracy.

The ResNet-18 implementation is based on the pre-trained model available at the official model zoo of Pytorch. It includes 21+2 layers: a convolutional layer with 7×7 kernels and stride of 2, a max-pooling layer with 3×3 kernels and stride of 2, 4 convolutional blocks with residual connections, each including 4 convolutional layers based on 3×3 kernels and strides

of 2 and 1, a 7×7 average-pooling layer with the stride of 7, and finally a 512×1000 fully-connected layer that provides the output prediction corresponding to 1000 classes. The network is trained on ~1.3M images of the ImageNet dataset for 150 epochs with a batch size of 256, the learning rate of 0.1 that is divided by 0.1 every 30 epochs (step scheduling), cross-entropy cost function, weight decay of 0.0001, and stochastic gradient descent optimization with a momentum of 0.9. The model achieves an average classification accuracy of ~70.2% tested on 50k images of the dataset. The networks are trained with 32-bit floating-point precision on Nvidia Titan X GPUs, and the learned parameters achieving the highest test accuracy are used as the baseline model.

In every model, the VMM operations are partitioned to nonoverlapping $N \times N$ kernels (see partitioning in general-purpose mixed-signal deep neural networks [67]). Similar to the VMM study, the obtained weights are mapped into target device conductances. The conductance tuning process for the constructed VMM kernels is then emulated using the dynamic model and previously discussed tuning algorithm. The imprecise tuned weights are then imported backed to the simulation setup. Subsequently, the inference tasks are performed on the generated models and evaluate the classification drop in each data point. For every case study, 12 model instances are generated (by using 12 sets of randomly generated switching threshold distributions). Fig. 14 shows the accuracy drop of running the inference test on both benchmarks versus the crossbar uniformity for various VMM sizes. Panel (a) and panel (b) correspond to ConvNet and ResNet-18, respectively. The box plot is obtained by simulating 12 random hardware instances (note that tuning simulations are extremely slow even when performed on a powerful server). The destructive impact of crossbar half-select disturbance is evident in both benchmarks, especially in ResNet-18 that performs the more complex

ImageNet classification. The trends are consistent with VMM simulations: The accuracy drops exponentially when the VMM size and normalized variations are increased. Notably, with 25% normalized variations and 64×64 crossbars, we achieve ~9% accuracy drop in the ConvNet and 18.5% on ResNet-18. In the next section, we introduce several methods, which restore this accuracy drop and improve the performance.

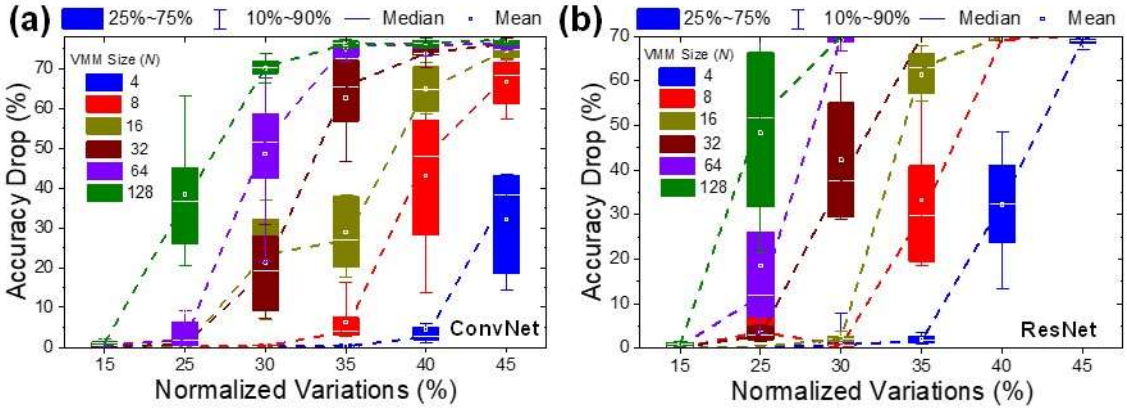


Fig. 14: The accuracy drop in deep neuromorphic networks versus crossbar uniformity: (a) ConvNet, (b) ResNet-18. Each box plot is obtained from the simulation of 12 different switching threshold distributions (which correspond to 12 chips). The dashed lines connect the median of the boxes.

2.3.5. Improving the Performance

The most straightforward solution to cope with the destructive impact of variations in the switching thresholds is to improve the fabrication process and device properties. The switching threshold variations in metal-oxide memristors depend on multiple factors. Forming voltage and current overshoot during the forming significantly contribute to device variability and can be tuned by a combination of oxide layer thickness and stoichiometry adjustment and optimized annealing conditions [24,119]. Here, we would like to focus on some post-fabrication techniques for mitigating the disturbance.

A. Hardware-Aware Training

In our recent works [118,120], imperfections of synaptic devices such as noise, temperature dependency, stuck-at fault, retention, and tuning error are compensated by the method of hardware-aware training: The training is performed fully ex-situ (no extra hardware cost), with the only subtle difference of including the device models and imperfections in the training phase for the purpose of generating more robust models. The simulation results of the previous section indicate that variations in switching thresholds lead to random tuning errors in the devices. Note that the tuning errors remain fixed during the inference, assuming devices have adequate retention. Nevertheless, tuning errors are chip-dependent, model-dependent, and unpredictable because of the intrinsic chip-specific distribution of switching thresholds. Despite that it is not feasible to predict and include the exact amount of errors in the training phase, the error distribution is predictable due to the uniform shape of weight distribution in a neural network model, but more importantly, because of using the same crossbar sizes and tuning algorithm (see modular accelerator architectures, e.g., aCortex [67]).

Modeling the tuning error during the training may increase the robustness of the trained model against half-select disturbance during the inference of the neural hardware. Note that this technique does not transform the shape of the tuning error distribution. Prior to computing the activation values in each update, the weights are converted to memristor conductances. Built-in uniform random number generator with the parameter ζ is then used to perturb conductances (both G^+ and G^- in the differential implementation). ζ should be optimized for a given network model and overall disturbance (which we now know is a function of VMM size, switching threshold variations, and target accuracy). After computing imprecise preactivations, the ideal weights are then restored before proceeding with the rest of the

training operations. Note that using a more complicated distribution for perturbation is also feasible though we suspect it will have a noticeable impact.

Fig. 15 shows the performance improvement achieved by this technique on the ConvNet benchmark implemented with 64×64 VMM blocks. The figure shows the accuracy drop versus the normalized variations for various values of ζ. The robustness of the deployed model is obviously increased with this method. For 15%, 25%, and 30% normalized variations, the optimum performance is achieved when ζ is set to 5%, 20%, and 30%, respectively. Notably, in the case of 25% normalized variations and 64×64 crossbars, the ~9% average accuracy drop is now reduced to ~1.87% using ζ=20%. The same trends of improvements are also observed in the case of ResNet-18 implemented with 64×64 crossbars. For example, using ζ=3% (20%) diminishes the accuracy drop from 18.5% to 3.5% (6.1%) for σ=25%.

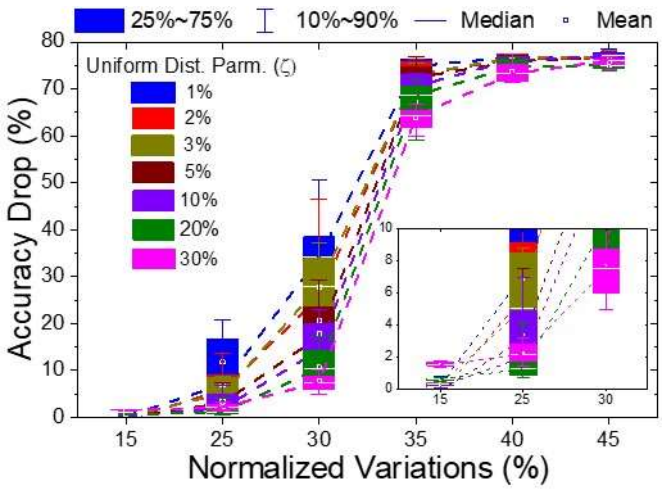


Fig. 15: Reducing the accuracy drop in ConvNet (64×64 VMMs) using the hardware-aware training technique. By emulating the distribution of the tuning error during the training, the network becomes more resilient toward the half-select disturbance. The inset shows the zoomed-in to the lower portion of the figure.

B. Improved Tuning Algorithm

In the previous chapter, we present a novel crossbar tuning procedure consisting of two methodologies for reducing the tail of tuning error distribution. First, the write voltage

amplitudes are limited to a specific voltage, which is decreased gradually within each tuning round. The consequences of restricting the maximum applied write voltage within each round are gradual reduction of net disturbance in each round and large (final) tuning error in high threshold devices. The former stems from the fact that low-to-moderate threshold devices become disturbed less and less as the tuning algorithm advances. The latter merely originates from the fact that the write voltage is not enough for high threshold devices to switch. In the second method, we initially identify devices with large set(reset) switching thresholds and switch them to the highest (lowest) conductive state prior to executing the first tuning round. Then, we take advantage of the possibility to encode the same weight with different target conductances in the differential pair implementation. In every round, when tuning a disturbed device with a threshold higher than the maximum voltage limit imposed by the first methodology, the state of the paired device is adjusted rather than the high voltage device. The application of these two novel techniques significantly reduces the tail of disturbed devices.

Fig. 16 demonstrates the effectiveness of using these novel tuning algorithms with and without applying the hardware-aware training technique. When no hardware-aware training is applied, the novel tuning algorithm reduces the accuracy drop regardless of the uniformity, predominantly when variations are higher than 20%. When the two techniques are both applied, the results are even better. A sub-percent accuracy drop is now feasible even with 30% normalized variations. For the notable case of 25%, the average drop now becomes insignificant when $\zeta = 2\%$ is used in the hardware-aware training. The simulation results of the ResNet-18 benchmark are also promising. For example, in the case of $N = 64$ and $\sigma = 25\%$, the improved tuning algorithm solely reduces the accuracy drop to 1.88%. Combined

with the hardware-aware training ($\zeta = 3\%$), we can decrease the accuracy drop to just 0.4%. In the initial simulation, we observe that the model generates almost random outputs (~70% accuracy drop) when the variations are $\sigma = 35\%$ and larger. While the two proposed techniques enable 6.9% and 17.2% accuracy drops, utilizing $\zeta = 20\%$ and $\zeta = 3\%$, respectively.

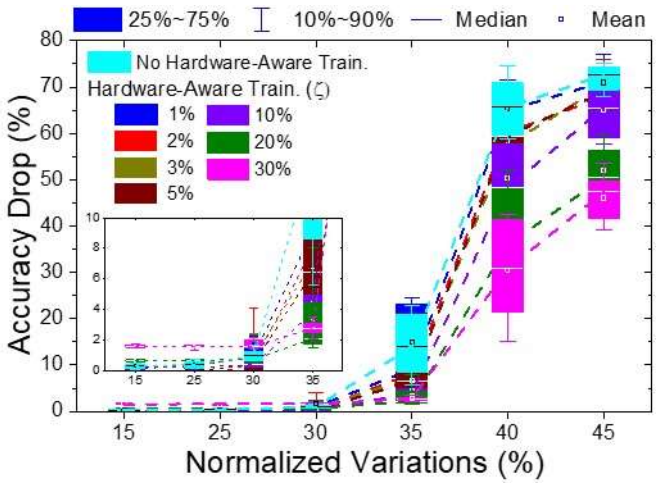


Fig. 16: Reducing the accuracy drop in ConvNet (64×64 VMMs) using the novel tuning algorithm with and without hardware-aware training. The inset shows the zoomed-in to the lower portion of the figure.

C. Modifying Switching Thresholds

Modifying the switching thresholds of outlier devices is another method for reducing the impact of variations in the switching thresholds. This correction process includes an unconventional continuous hard reset operation, which pushes the outlier device close to its virgin state, followed by a voltage-controlled reforming procedure, which revives the device with slightly shifted switching characteristics. Our experiments show that the correction process results in a stochastic shift in the switching threshold of devices, which means the refreshed device could have improved switching properties. Applying this technique to outlier devices (that feature low voltage thresholds) reduces the spread of variations, which in turn improves the accuracy of the implemented model.

Fig. 17 shows the results of the experiments developed to validate this idea. First, a virgin device in the crossbar is formed and tuned to 50 kΩ. Then, its switching thresholds are measured by tuning it repeatedly to 100 kΩ and 10 kΩ. After 10 rounds, the device is hard reset to >1MΩ and then revived and tuned to 50 kΩ. Switching thresholds are measured again in a similar fashion. The process is performed one more time just to make the results more illustrative. Fig. 17a shows the experimental results, and the inset of Fig. 26a shows how the switching thresholds are changed after the initial forming (i.e., the original thresholds) and each reviving round. The stochastic shift of the switching thresholds is quite obvious from these data. In addition, a one-time switching threshold modification is applied on 60 devices with various initial switching thresholds, and Fig. 17b corroborates the stochastic nature of this shift in the symmetric histogram of relative change in the switching threshold after the modification process is applied. Specifically, as an example of modifying an outlier device, we apply the modification process to a device with an average set and reset switching thresholds of +0.8 V and -0.8 V, respectively (among ten switching rounds). Then, it is observed that the refreshed device has an average set voltage of 0.95 V and an average reset voltage of -1.2 V, which are significantly better and closer to the typical average switching thresholds of the crossbar. Due to the limitations of our experimental setup, we can not validate the impact of this method with direct system-level experimental results. On the other hand, simulations of reducing the variations in the benchmarks are trivial, and its effects are obvious. Therefore, the study of the impact of this technique on large-scale neuromorphic architectures is considered as a future work when integrated 0T1R memristive systems are available.

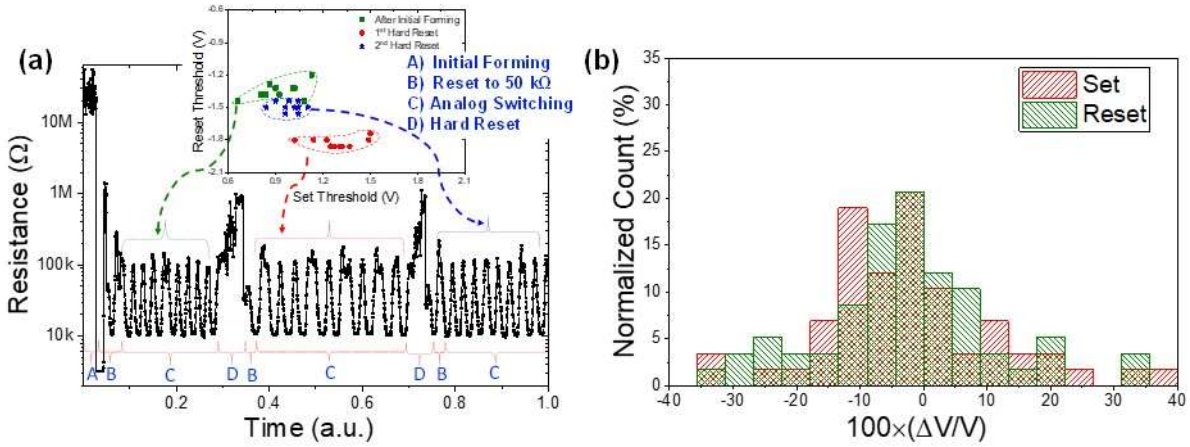


Fig. 17: Modulation of switching thresholds: (a) The experimental results of the modification process applied on a virgin device in the crossbar. The process includes initial electroforming, tuning the device to 50 kΩ, analog switching for ten times between 10 kΩ and 100 kΩ, which are close to typical the lowest and highest resistance states, and hard reset to a state of more than 1 MΩ. The inset shows the switching thresholds measured in each round. (b) The histogram of the stochastic relative change ($100\times\Delta V/V$) in the average set/reset switching thresholds (V) for 60 devices after performing the hard reset and revival processes on them.

2.3.6. Discussion

The findings in this section suggest that the prospects of using 0T1R crossbars in neuromorphic computing are high. In the general VMM study, we uncover exciting opportunities and interesting tradeoffs about these circuits for the first time: 1) The relationship between the computational error and the crossbar size, uniformity, and target tuning error is thoroughly investigated. 2) We present the periodic and instability of tuning error in large nonuniform crossbars in addition to the linear and exponential dependency of computing accuracy to uniformity at small, moderate, and large VMM sizes. 3) It is shown that in large VMMs, very precise tuning of devices requires a large number of pulses, which in turn may lead to more disturbance and reduction of the ultimate computing accuracy. 4) Slight increase in the computational error is inevitable in very large 0T1R crossbars even with zero variations since even a small half-select voltage drop could become potentially noticeable when the total number of pulses grows very large. 5) We compare the computational accuracy

of state-of-the-art 1T1R (~3% target error reported in [13] based on extremely conductive devices) and 0T1R crossbars (1% target tuning and σ~25% reported in [113]) and report similar computing accuracy when the 0T1R crossbars are tuned with 1% target precision or worse by only ~1% when using the same (as 1T1R) tuning precision of 3%.

Evidently, such results suggest excellent prospects of using 0T1R circuits in building large neuromorphic networks since passive technology offers much better density than 1T1R technology. To clarify this argument, we study the area consumption of 64×64 standalone 1T1R and 0T1R circuits that include the memristor arrays, programming switches, and read switches. Note that the programming switches facilitate the forming and write operation of the crossbar and virtually implement the V/2 scheme and the read switches decouple the programming circuit from the analog peripheries (e.g., data converters and sensing circuits). The 1T1R circuit obviously includes an extra selector switch for every memristor.

We assume a scaled version of the 1T1R technology since the state-of-the-art demonstration (with properly reported statistics) [13] is based on extremely large cell size and conductance. In other words, similar device parameters (e.g., G_{on} and G_{off}) are used between the two topologies for a fair comparison. Analog switches are also designed and laid out in a 65 nm CMOS process. Based on these considerations, we design both 0T1R and 1T1R circuits based on 2 types of (thin-oxide 1.2 V and thick-oxide 3.3 V) switches and taking the midrange conductance of devices and forming current as variables. Such information reveals the trends of scaling the device size (which should result in lower midrange conductance due to reduced leakage) and scaling the switching/forming voltage (which is an important near-future goal in designing the next generation of both 0T1R and 1T1R memristor circuits).

The area of the read switches is the same between the two topologies and depends on the maximum read current and tolerable drop voltage on them. Here, the on-conductance of read switches is designed to ~$100G_{\mathrm{on}}$, which is selected based on numbers from the ResNet-18 implementation on aCortex architecture. Note that since memristors are linear in the read voltage regime, it is reasonably straightforward to compensate for the voltage drop on the read switches by adjusting the state of the devices. The sizing of programming switches is decided based on the worst-case write current drawn from an electrode during the forming operation. In 0T1R circuits, the write current is the sum of the current used to form a device and the leakage current in the rest of the crossbar (since other electrodes are floated). The maximum I_{wrt} is drawn from the circuit when electroforming the last device. Note that, to minimize the leakage, we always reset all previously formed devices to G_{off} during the forming operation. In 1T1R circuits, I_{wrt} is the current needed for forming a single device. In all circuits, the number of fingers of MOSFETs is adjusted to minimize the area.

Fig. 18 shows the circuit area offered by 1T1R and 0T1R topologies in different cases. Specifically, it is shown that by scaling the device conductance, the circuit area reduces dramatically for 0T1R topology because of reduced leakage (specifically for small scaling factors in which the worst-case write current is overwhelmingly dominated by leakage). The gradual circuit area reduction in 1T1R is due to the diminished area of read switches. Scaling forming current is though more noticeable in 1T1R because the selectors which dominate the area are shrunk. Scaling forming/switching voltage is effective and essential in both technologies, which would allow the use of scalable thin-oxide transistors for programming switches. Fig. 18 also indicates that 0T1R circuits outperform 1T1R designs with a tremendous ~5× area gain if we use the current state-of-the-art analog-grade memristors (S=1,

~250 µA forming, and ~3.3 forming/switching voltage). By scaling all functional parameters of memristors (to (S=10, ~50 µA forming, and ~1.2 forming/switching voltage), the area gain becomes ~7×, signifying outstanding prospects for passive memristive circuits. For larger crossbars, the gain becomes even larger. In future works, we will study the impact of 3D integration of multilayer 0T1R circuits and the possibility of sharing programming switches, which should enlarge the gain even further. Note that the scaling results are relevant as long as the switching characteristics and device properties remain fine.

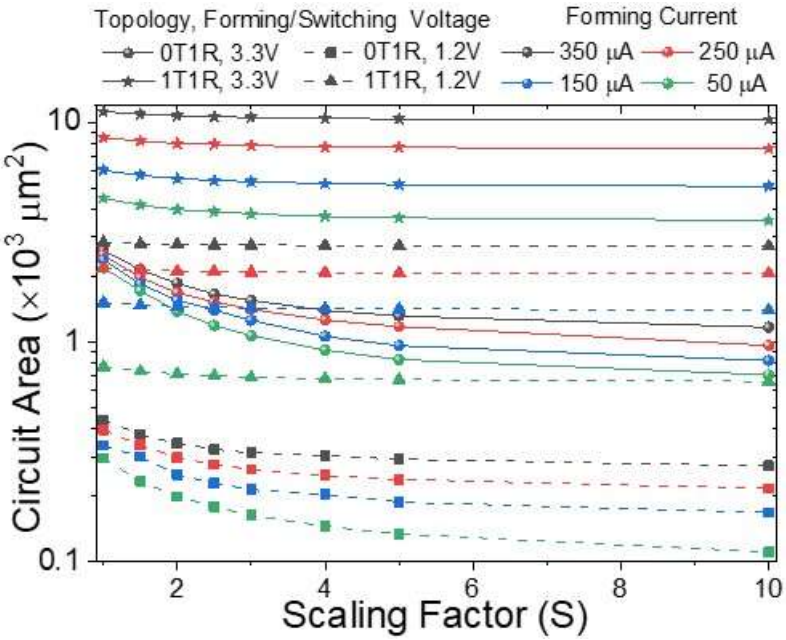


Fig. 18: Circuit (crossbar + switches) area of 64×64 0T1R and 1T1R topologies versus the scaling factor (S) for various forming currents and voltages based on a 65 nm CMOS process. For S=1, we have $G_{\text{on}} = 100$ µS and $G_{\text{off}} = 8.3$ µS. In the cases with S >1, both G_{on} and G_{off} are scaled with S. When forming/switching voltage is 3.3 V, thick-oxide 3.3 V transistors are used to implement all switches and the maximum of 0.7 voltage drop is allowed on programming switches (i.e., the maximum input voltage is 4 V). When forming/switching voltage is 1.2 V, thin-oxide 1.2 V transistors are used to implement all switches and the maximum of 0.3 voltage drop is allowed on programming switches (i.e., the maximum input voltage is 1.5 V).

Three techniques are explored for mitigating the impact of nonuniform *IV* characteristics of 0T1R memristors in neuromorphic circuits. The simulation results indicate that these techniques enable software-equivalent accuracy on both ResNet-18 and ConvNet

benchmarks, in the case of N=64 and 25% normalized variations, which corresponds to the features of our recent fabricated crossbar. In addition, the presented data in Fig. 16 suggest that a sub-percent accuracy drop is achievable in advanced neuromorphic circuits with even ~30% normalized variations using 64×64 crossbars, which leads to a balanced resource utilization at system levels, as promised by theoretical architectural studies [67].

Let us also mention several limitations of these mitigation techniques. First, hardware-aware training is not a viable option in some neuromorphic tasks, e.g., neurooptimization [8], in which the weights are fixed and predetermined by some constraints of the applications. In such cases, the practical solutions are improved tuning algorithm, fabrication process, outlier correction, and, if needed, reducing the crossbar dimensions. Second, the switching threshold modification method should only be used for outlier devices once or a few times to prevent damaging the devices or reducing their endurance life.

Finally, here we focused on the uniformity that is the primary challenge of upscaling 0T1R crossbars. Another challenge facing further progress is CMOS compatibility. Some technologies require high-temperature epitaxial growth [87], which is incompatible with monolithic CMOS or three-dimensional integration. Some processes use non-CMOS compatible materials, e.g., silver, gold [105], etc., and most previous works use nonstandard deposition techniques such as liftoff. Often, the electroforming and switching voltage range in (both 1T1R and 0T1R) technologies are higher than the core voltage (and sometimes IO level) of standard sub-micron CMOS processes. Fig. 18 suggests downscaling device conductance, forming current, and switching voltages must be followed as an important future direction, as it leads to a dramatic enhancement of density in both analog 1T1R and 0T1R circuits.

In summary, the excellent scalability prospects of memristors are promising for designing energy-efficient and compact neuromorphic circuits. However, the strict uniformity requirements on the *IV* characteristics of memristors challenge the upscaling of 0T1R memristor crossbars. In this section, we have conducted an in-depth analysis of this problem and studied the tradeoffs between computing accuracy, crossbar size, switching threshold variations, and target precision. First, we have demonstrated our results using the benchmark of general matrix multiplication operations and uncovered interesting tradeoffs and exciting opportunities in these circuits. Then, study the impact of crossbar uniformity in two representative deep neural networks and explore three solutions (hardware-aware training, improved tuning algorithm, and switching threshold modification) for improving the performance. More importantly, it is shown that the current state-of-the-art analog-grade 0T1R technology can offer software-equivalent accuracy of advanced deep neural networks. Finally, we have analyzed the density prospects of memristive circuits and showed at least ×5 superior density of 0T1R against 1T1R circuits. Further, we have shown that downscaling the device characteristics such as midrange conductance will increase this superiority even further.

2.4. IR-drop and Nonlinearity Analysis in Memristive Circuits

Device nonlinearity and interconnect parasitics are two important factors that limit the computing precision of memristor-based VMMs. This chapter investigates the impact of such non-idealities in analog current-mode memristive VMMs through simulations and experiments on the most prospective passive crossbars. To make our analytical study more general, we target two applications of VMM circuits: an analog multi-layer perceptron implementation and 2D convolution for edge detection filtering. Several works in the

literature addressed similar topics, but with a focus on devices with selectors [81,98], write operation [121], or programming error due to variations. Nonlinear input mapping is proposed in [122] to compensate device nonlinearity. However, variations and interconnect parasitics are neglected in that study. Only linear devices and the impact of IR drop are considered in [123]. Passive crossbar circuits are explored in [124] using a simplified model focusing only on nonlinearity and IR drop. Similarly, Ref. [125] proposes a conversion algorithm to minimize the error due to device non-idealities and IR drop. Finally, some works explored the impact of non-idealities in the context of specific applications, such as neuromorphic computing [126,127], which is typically much more resilient to precision errors or digital memory [128].

We model $N \times N$ crossbar circuits with passively integrated TiO_{2-x} memristors (Fig. 19) based on the technology developed by our group. Our general focus is on the circuit and device parameters, e.g., values of N, wire conductance g_{wire}, and memristor currents, specific to our technology. Note that the considered TiO_{2-x} devices have rather typical I-V characteristics of many metal-oxide memristors. We also extend our analysis to larger g_{wire} values, representative of more advanced fabrication processes (e.g., with larger aspect ratio crossbar electrodes made of higher conductance metals, compared to those used in our circuits). Hence, we believe that the results of this paper are quite general.

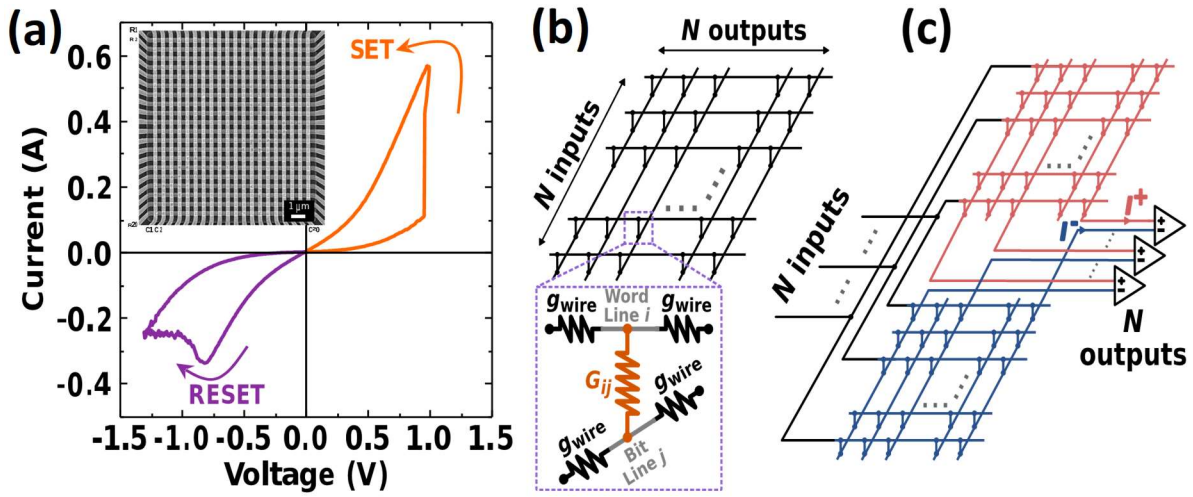


Fig. 19: Experimental *I-V* curve of a TiO_2 memristor. Inset shows a scanning electron microscopy image of a 20×20 passive crossbar circuit. (b) Single-quadrant and (c) differential $N \times N$ crossbar circuit with memristive crosspoint devices.

Simulation results are obtained from SPICE using an in-house phenomenological compact model of TiO_2 memristors [115]. As opposed to previous work, this compact model is developed based on an extensive statistical study and captures the device imperfections needed for a comprehensive analysis of memristive circuits and systems. The studied non-idealities include nonlinearities in static *I-V* characteristics, device-to-device variations, noise, and temperature dependency (e.g., see Fig. 20).

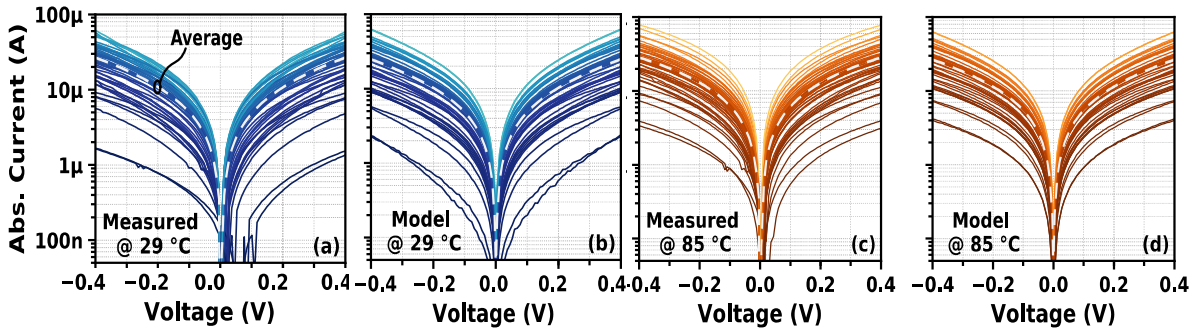


Fig. 20: Example of *I-V* curves for 50 devices: (left) experimental data, (right) prediction by the model assuming similar conductance at 0.1V.

We consider conventional memristor-based VMM topology in which data are encoded in voltage amplitude of signals. Compared to time-based encoding, the voltage-mode approach is fully compatible with the most prospective passive technology and is potentially more

promising, in terms of throughput and energy efficiency, particularly in medium precision regimes – see, e.g., our previous works on system-level analysis [16,129]. We study both single-quadrant (Fig. 19b) and differential (Fig. 19c) topologies and take into account parasitic wire resistance of memristive crossbar circuits. For each crossbar size N, we randomly generated 512 input voltage vectors, with vector elements uniformly distributed in the range $[0, U_{\max}]$ - see Table II for the definition of all parameters used in this study. In addition, 512 crossbar circuits are randomly generated, each with unique device-to-device (d2d) variations and crosspoint conductances (at 0.1 V), uniformly distributed in the range $[G_{\min}, G_{\max}]$. The crosspoint conductances were obtained indirectly by first generating dimensionless weights and then converting them according to the VMM topology (Table II). The VMM errors were then calculated for all combinations of input vectors and crossbar circuits, with a total of 256k configurations for each N. Most of the results are reported in terms $|\varepsilon|_{99.9\%}$, which is 99.9% percentile of output current errors $|\varepsilon|$ for each studied crossbar size N, where $|\varepsilon|$ is the absolute difference between ideal and the actual output current, normalized to its maximum value.

Table II. Notations (top) and related equations (bottom).

N	Linear dimensions of the crossbar array
B	Number of connections used for bootstrapping per crossbar line
$U_{\max}$	Largest (non-disturbing) voltage applied to the crossbar circuit inputs during operation
$kU_{\max}$	Voltage at which device conductance is tuned
g_{wire}	Crossbar line (full-pitch- long segment) conductance (Fig. 1b)
G_{ij}	Crosspoint device conductance (Fig. 1b)
$G_{\min}, G_{\max}$	Minimum, maximum value of G_{ij}
$I^{(n)}$	n-th element of the output current vector
$I^{(n)}_{\text{ideal}}$	Ideal value for $I^{(n)}$, i.e., for linear devices and $g_{\text{wire}} \to \infty$
$\lvert I\rvert_{\max}$	Maximum absolute current value
ε	Relative (computing) current error – see below
$\lvert\varepsilon\rvert_{99.9\%}$	99.9% percentile of $\lvert\varepsilon\rvert$, also referred to as worst-case error
W_{ij}	Weight value, assumed in [0, 1] and [-1, 1] ranges for single-quadrant and differential topologies, respectively
Single-quadrant	$G_{ij} = G_{\min} + W_{ij}(G_{\max} - G_{\min})$
	$\varepsilon = (I^{(n)} - I^{(n)}_{\text{ideal}}) \,/\, \lvert I\rvert_{\max}$
Differential	$G_{ij} = G^{+}_{ij} - G^{-}_{ij}$ $G^{\pm}_{ij} = G_{\min} + (1 \pm W_{ij}) \times (G_{\max} - G_{\min})/2$
	$\varepsilon = (I^{(n)} - I^{(n)}_{\text{ideal}}) \,/\, (2\,\lvert I\rvert_{\max})$

2.4.1. Computing Precision Analysis

In our first study, we consider g_{wire} = 0.4 S, which corresponds to the measured line conductance in 20×20 crossbar circuits [15,16] (but smaller compared to other recent works [2]). Fig. 21 shows histograms of the simulated output currents for each output of a 16×16 VMM circuit for different topologies, biasing strategies, and temperatures. The single-quadrant architecture severely suffers from the voltage drop on interconnect parasitics. Biasing the word lines from both sides can help to mitigate the shift and the spread of the errors with respect to the output index (Fig. 21a, b) at the temperature (25 °C) used during programming. (A more general and powerful biasing approach is further discussed in Section IV.) Nevertheless, it does not reduce the sensitivity to the temperature (Fig. 21c, d). The differential scheme (Fig. 21e-i), however, centers and narrows down the distribution of output errors around zero. This is because the effective conductance of the device is (almost) a monotonic function of temperature. Hence, the temperature-related terms cancel out in the differential topology. This topology is particularly appealing to keep the error as low as possible when combined with two-sided voltage biasing. Despite the conservative choice of g_{wire}, $|\varepsilon|_{99.9\%}$ is still below 1.5% and 0.75%, for 85 ˚C and 25 ˚C, respectively (Fig. 21f, i).

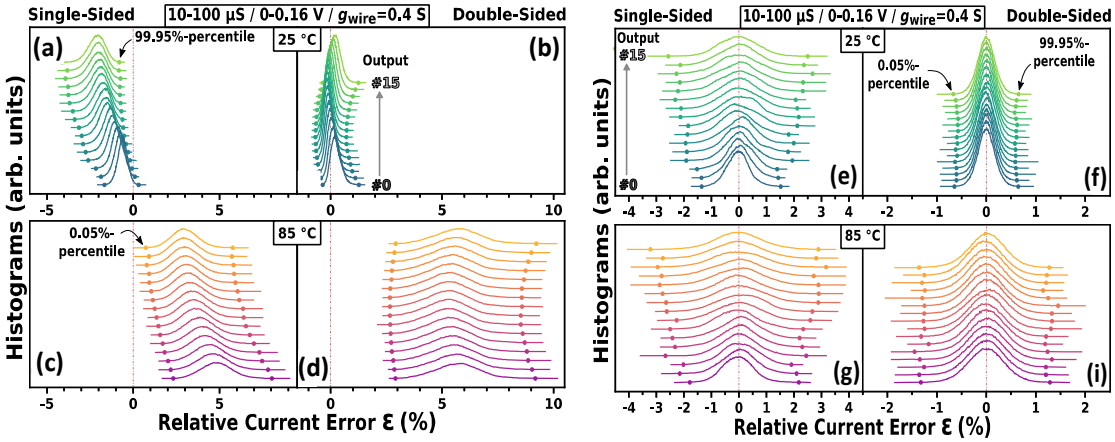


Fig. 21: Simulated results for output current error ε for (a-d) single-quadrant and (e-i) differential topologies implementing 16×16 VMM. Output #0 (#0 and #15) is the closest to

where the input voltages are applied to single-quadrant (differential) topology. Panels (a, c, e, g) are for single-sided voltage biasing, while (b, d, f, i) are for double-sided ones. The top and bottom panels are for 25 ˚C and 85 ˚C, respectively. For all cases, G_{min} = 10 µS, G_{max} = 100 µS, and U_{max} = 0.16 V.

For a more practical case of larger crossbar circuits, the finite g_{wire} can cause significant voltage drops, reducing the effective voltage drop on the crosspoint device (Fig. 22a). At smaller g_{wire}, the $|\varepsilon|_{99.9\%}$ is roughly inversely proportional to the square of g_{wire}, which is consistent with the worst-case error due to the IR drops on the crossbar lines. At larger g_{wire}, $|\varepsilon|_{99.9\%}$ is leveling, independent of the voltage and the conductance ranges (Fig. 22b). In fact, $|\varepsilon|_{99.9\%}$ slightly increases before plateauing due to excessive currents injected by nonlinear devices at higher biases, which compensates for the current deficiency created by IR drops on the crossbar lines. In addition, intrinsic device characteristics, e.g., the higher temperature sensitivity in lower conductance ranges, also have a significant impact on the error.

Furthermore, the error plateau is lower for smaller N (Fig. 22b inset). Assuming g_{wire} > 1 S similar to [2], Fig. 22b results predict $|\varepsilon|_{99.9\%}$ < 1% at 25 ˚C, even for large, >1K-cell crossbar circuit. Unsurprisingly, g_{wire} should be increased quadratically with N to ensure error below 1% (Fig. 22c). Also, the differential topology is naturally more immune to IR-drop because its impact on the currents through both devices is compensated by a differential pair.

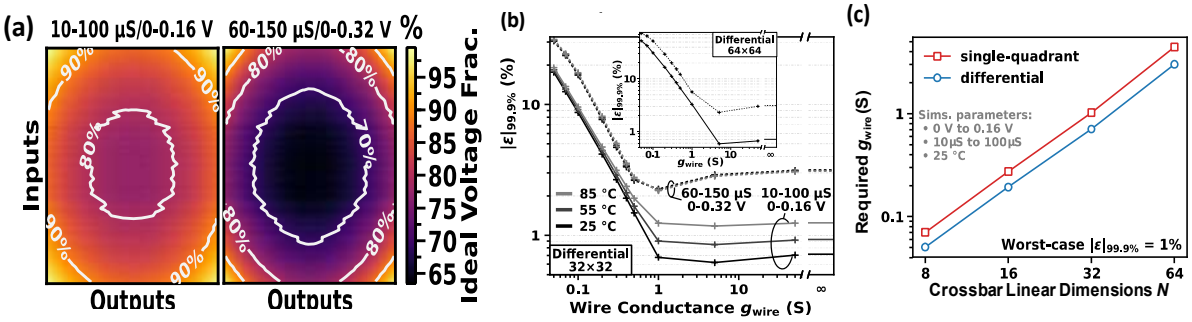


Fig. 22: Heatmap of normalized (by U_{max}) average voltages across crosspoint devices in a 64×64 crossbar circuit with g_{wire} = 0.4 S and double-sided voltage biasing, simulated at room temperature (25 ˚C), for two different values of U_{max} and ranges of device conductances. (b) Simulated worst-case error as a function of g_{wire} for differential 32×32 circuit. Inset shows the

results for 64×64 differential crossbar circuit simulated at room temperature. (c) Extrapolated wire conductance g_{wire}, which is required to ensure the 1% worst-case error.

For relatively small crossbar circuits with smaller voltage drops, device nonlinearity becomes the main precision-limiting factor, while for larger crossbar circuits, the dominant factor is the interconnect parasitics (Fig. 23). On the other hand, device-to-device variations do not have much impact on the error, at least for the considered applications. This is because a write-verify tuning algorithm allows the program to accurately program the device conductance at read voltage despite variations in *I-V* characteristics. Furthermore, the impact of device-to-device variations reduce even further for large crossbars due to averaging

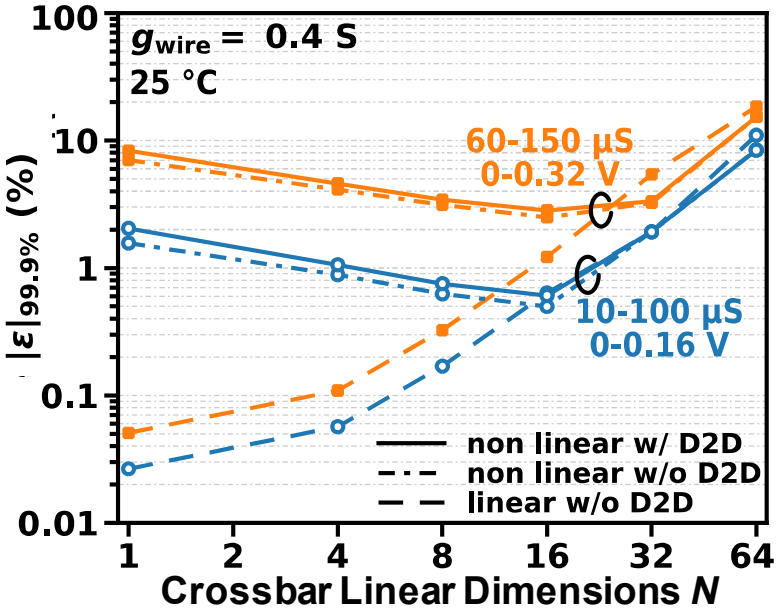


Fig. 23: Impact of *I-V* static nonlinearity and d2d variations on the worst-case error for differential topologies.

The computing error due to nonlinearity in the static *I-V* characteristics can be reduced by optimizing the procedure of mapping weights to the memory states of crosspoint devices. In the case of ideal, linear devices, the slope of *I-V* curve, i.e., the memory state, is typically assessed by measuring device current at the highest operating voltage (U_{max}), when using a write-verify tuning algorithm. For the devices with nonlinear *I-V* curves, such an approach

leads to a negligible error at U_{max} voltage input, but the error might be significant at voltages below U_{max} due to smaller effective conductance. To reduce such error, we can tune the crosspoint devices at smaller voltages kU_{max}, where $0 < k \leq 1$, i.e., by setting the device's effective conductance at $I(kU_{max})/(kU_{max})$ to the desired value at the tuning algorithm. In this case, the error would be the smallest at kU_{max} and is more balanced between the larger and smaller ranges of the input voltages (Fig. 24a inset). For the in-house devices, G = 100 µS, and the distribution of the inputs assumed in the modeling (Section II), the computing error is minimized by using $k \approx \text{sqrt}(2)/2 \approx 0.71$ (Fig. 24a). Fig. 24b shows a comparison of VMM output currents from crossbar circuits with ideal (linear) devices and those with nonlinear devices for two cases of k. As expected from Fig. 25a results, the currents are higher for k = 0.5 due to larger error integral at higher input voltages. The distribution of currents for the case of nonlinear devices is almost perfectly matching the ideal one when using optimal k. This is because individual current errors of single devices (i.e., those in computed product terms) are canceling each other out when device currents are summed up on the crossbar lines.

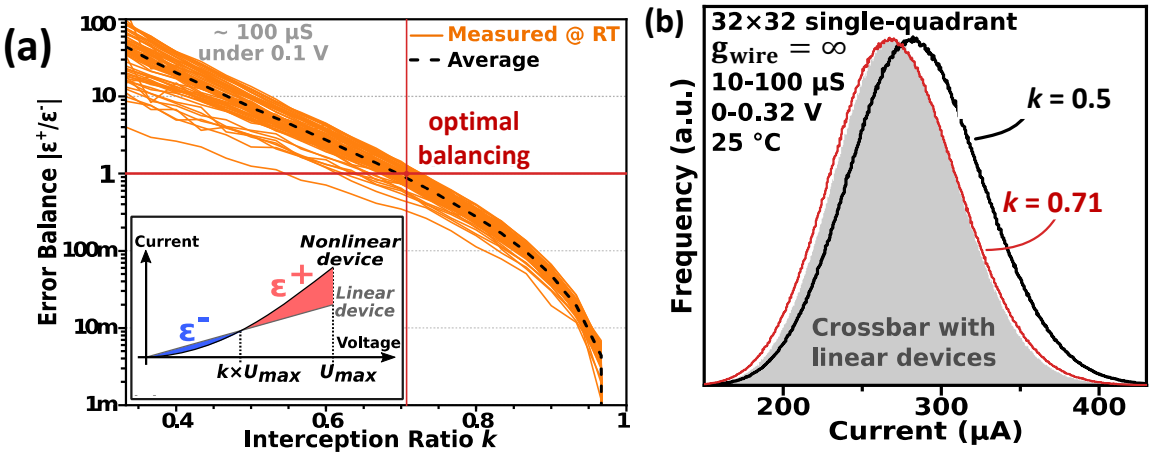


Fig. 24: The ratio between positive and negative error integrals, calculated from the measured *I-V* characteristics of TiO_2 devices, which were tuned to have $I(kU_{max})/(kU_{max})$ = 100 µS at 27˚C, as a function of used k value. Inset shows positive (red) and negative (blues) current error integrals for single devices schematically. (b) Simulated output currents in a 32×32 single-quadrant architecture with nonlinear crosspoint devices, when tuned at k = 0.5 (black line) and k = 0.71 (red line), and ideal linear devices (grey-filled area).

The error balancing technique works well when voltage drops on the crossbar lines are not significant. However, the error can be even larger otherwise compared to the suboptimal balancing approach, e.g., for a 32×32 circuit with g_{wire} = 0.4 S (Fig. 25). This is because IR drops across crossbar lines compensate higher currents for the suboptimal balancing (i.e., the right shift of the currents in the histogram in Fig. 24b). One solution to deal with large IR drops is to compute optimal values of k based on the particular device location in the crossbar, e.g., by combining the balancing technique with the one described in Ref. 125.

An orthogonal solution is to employ a bootstrapping technique, e.g., similar to the one used in NOR flash memory circuits. In a bootstrapped design, all crossbar lines are backed up with spare lines, which, e.g., can be routed in the lower metal layers for back-end-integrated crossbar circuits. Each spare line is connected to the original crossbar line in $B > 1$ locations (denoted as "B×-bootstrapping"), which are equally distributed along the length of the line. For example, $B = 2$ implies that the original and spare lines are connected at the edges of the crossbar, i.e., corresponds to the already mentioned double-sided architecture. For 3×-bootstrapping, there are three connections - one in the middle and two at ends of the line, etc. Bootstrapping technique significantly improves the computing precision (Fig. 25a) while comes at the typically acceptable cost of utilizing additional metal layers below or above the crossbar array. For passive memristor technology, bootstrapping also requires increasing crossbar dimensions from N to $N+B-2$ to accommodate connections inside the crossbar array, though such overhead is minor for the most practical cases $N >> B$.

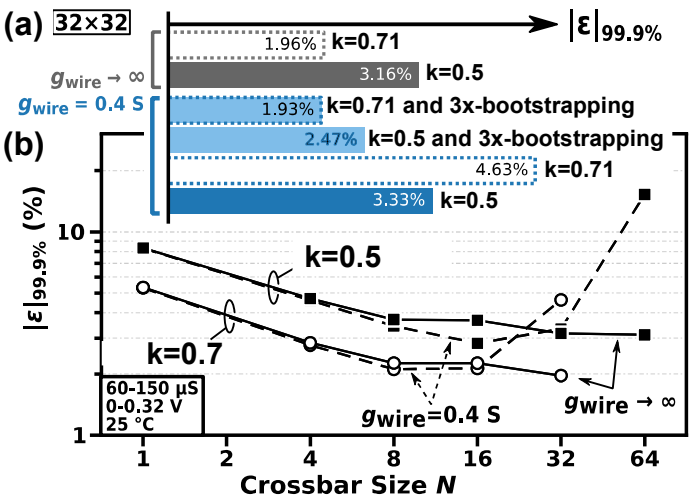


Fig. 25: Worst-case error for (a) 32×32 crossbar circuit and (b) as a function of crossbar dimension when using various techniques for improving precision and two assumptions of wire conductance. Wire resistance of the spare lines is neglected in the analysis of the bootstrapped circuits.

2.4.2. Application Demonstrations

The proposed techniques for improving precision are further verified by modeling two representative applications of mixed-signal current-mode VMM circuits. The first studied application is edge detection with 5×5 Laplacian of Gaussian filter, in which convolution of an image with a specific filter is computed to extract high-frequency information or image edges (Fig. 26). The image convolution operation is modeled assuming differential architecture with 25 inputs and 1 output for the specific image (Fig. 26a) using a hybrid approach. In particular, 50 devices in a 20×20 crossbar circuit are tuned to the desired values corresponding to the kernel weights (Fig. 26b) at the voltages specific to the used k, and their static I-V characteristics are collected. The data are then fitted using the approach discussed in [25] and used for simulating dot-product currents. 8 different implementations with different k, B, and g_{wire} are studied (Fig. 26e inset), including ideal case scenario, i.e., with $g_{wire} = \infty$ and linear I-V characteristics. Fig. 26c shows an example of filtered image assuming scenario D, i.e., using measured I-V characteristics, $k = 0.5$, $B = 2$, and $g_{wire} = 0.4$ S.

The results show that due to smaller parasitics, the crosspoint device nonlinearity is a major source of computing error, see, e.g., scenario A vs. B (Fig. 26d, e). This is why the balancing technique is the most useful for this application. Indeed, among the considered nonlinear device scenarios B, D, F, G, H, the error is smaller for scenarios F, G, H. On the other hand, bootstrapping does not help and can actually increase error (e.g., E cf. B, and F cf. H). This is due to the already mentioned compensation of IR drops across the crossbar. Even $k = 0.71$ is apparently not optimal (and hence H has a smaller error than F) for this particular application because of different distributions of conductances and inputs as compared to those used in Figs. 2.24 and 2.25.

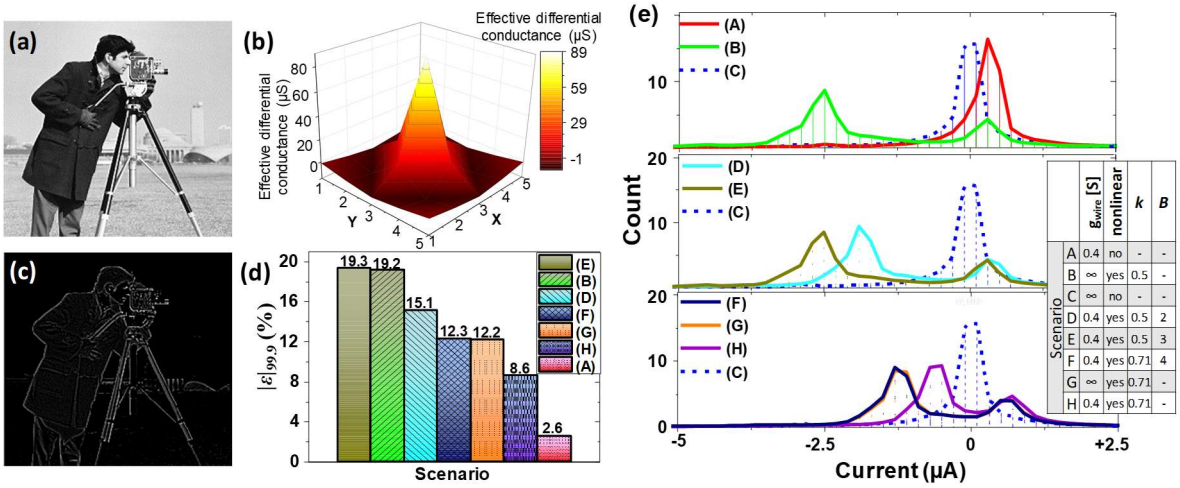


Fig. 26: Modeling of edge detection algorithm using 5×5 Laplacian of Gaussian filter assuming differential implementation based on two 25×1 memristive crossbar circuits and taking into account device's *I-V* nonlinearity and d2d variations: (a) Original image, (b) effective conductance of a differential pair used to implement a 5×5 filter. X and Y are filter dimensions. (c) simulation results for the computed image assuming 2×bootstrapping and g_{wire} = 0.4 S. (d) The worst-case error and (e) output current histograms for several considered scenarios. The details for each studied scenario are provided in the inset of panel e. T = 25˚C, U_{max} = 0.16 V.

The second studied application is the neuromorphic inference of MNIST benchmark images using a 784-64-10 multilayer perceptron classifier with rectified linear activation (Fig. 27). The first layer is modeled by assuming that 24 64×64 and 2 17×64 crossbar circuits are connected in two 785×64 virtual crossbars to realize differential architecture, while the second

layer is modeled with two 65×10 crossbar circuits. (The additional inputs are due to the bias.) The other hyperparameters and ex-situ training approach (with 60k/20k training/test images) are similar to [70]. The inference is simulated using the memristor compact model, which accounts for d2d variations in *I-V* characteristics [115]. It also assumes that input voltages for physical crossbar circuits are applied individually (i.e., that $N \leq 64$). The computing error in the first MLP layer (error in the output currents) and the corresponding classification errors are shown in Fig. 27c and d, respectively, for several scenarios (Fig. 27e). The results show that, unlike for previously studied applications, the impact of IR drops on the performance is more severe compared to device nonlinearity (test 2 cf. tests 1 and 3). This is due to smaller devices' conductances (i.e., large number of small weights as shown in Fig. 26b) as well as larger crossbar circuits. Both VMM error and the classification accuracy improves by increasing the crossbar line conductance (tests 4, 6, 7, 8, 9) or the number of bootstrapping connections (tests 6, 10, 11). Similar to the previous application, a small non-zero wire resistance could be beneficial for compensating current overshoot (test 3 cf. test 9). The results also show that the error is the largest for the single-sided architecture (test 4), for which only half of the crossbar circuits are employed in modeling, while a combination of more optimal balancing and aggressive bootstrapping leads to the classification performance of 2.09%. This number is close to the best-case 2%, obtained by simulating the same MLP network in software using high precision arithmetic – see, e.g., test 13 cf. test 1.

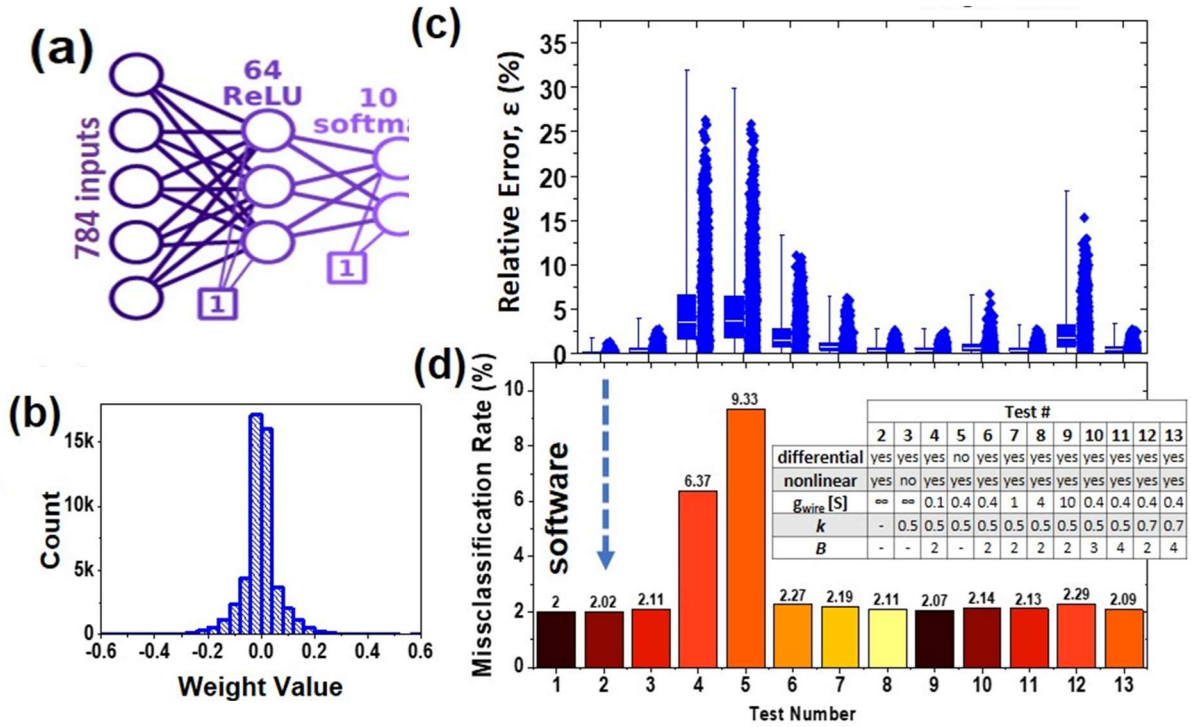


Fig. 27:Modeling of image classification inference with multilayer perceptron network: (a) Studied network. (b) Histogram of VMM weights for classification of MNIST benchmark images, obtained using ex-situ training method. (c) Simulated error in the output currents of the 1st layer VMM circuits and (d) corresponding misclassification errors for several studied scenarios (tests). The details for each studied scenario are provided in the inset of panel d.

In summary, we have developed a framework for circuit-level simulations of memristive crossbar circuits and utilized comprehensive device models as well as experimentally measured data for metal-oxide memristors to investigate the impact of various imperfections on computing precision of analog memristor-based VMM circuits. Using statistical numerical simulations, we quantified the impact of interconnect parasitics and analyzed different topologies on the precision under the range of temperatures. Finally, error balancing and bootstrapping techniques are proposed to mitigate device and circuit imperfections, which are further verified by modeling two representative applications.

2.5. Low-Power Sensing Circuit for Current-mode VMMs

Peripheral (sensing) circuits are perhaps the most energy and area-consuming components in current-mode VMMs. For example, the power consumption of the peripheral circuitry exceeds 90% in [70] and 83% in [125]. The reported area overhead is crudely 95% and more than 55%, respectively, for these two studies. As a result, the design of an energy-efficient neuromorphic circuit hinges on the structure of peripherals. Here, we design a very efficient topology for the current sensing in current-mode VMMs. Specifically, we focus on redesigned 55-nm ESF3 NOR flash memory (Fig. 28), which offers very high output impedance due to its split-gate structure. The experimentally measured output resistance is about >10 GΩ in the subthreshold regime for the targeted current range. Also, the compact cell structure results in a very low capacitance, of the order of ~75 aF/cell on average, in the subthreshold regime. (More details on the various aspects of this technology, including *IV* characteristics, erasure, and programming operation, cycling endurance, retention, noise, are discussed in [40].) Figure 28c shows the common gate-couple topology, which effectively performs the dot-product of a current-encoded input vector by dimension-less predetermined weights (programmed to flash memories) to generate a current-encoded output vector. The input current vector is applied to an array of diode-connected floating gate memory cells in this design. The two-quadrant multiplication is implemented by dedicating two rows per output and using the conventional differential weight scheme to sense and subtract the currents from these two rows.

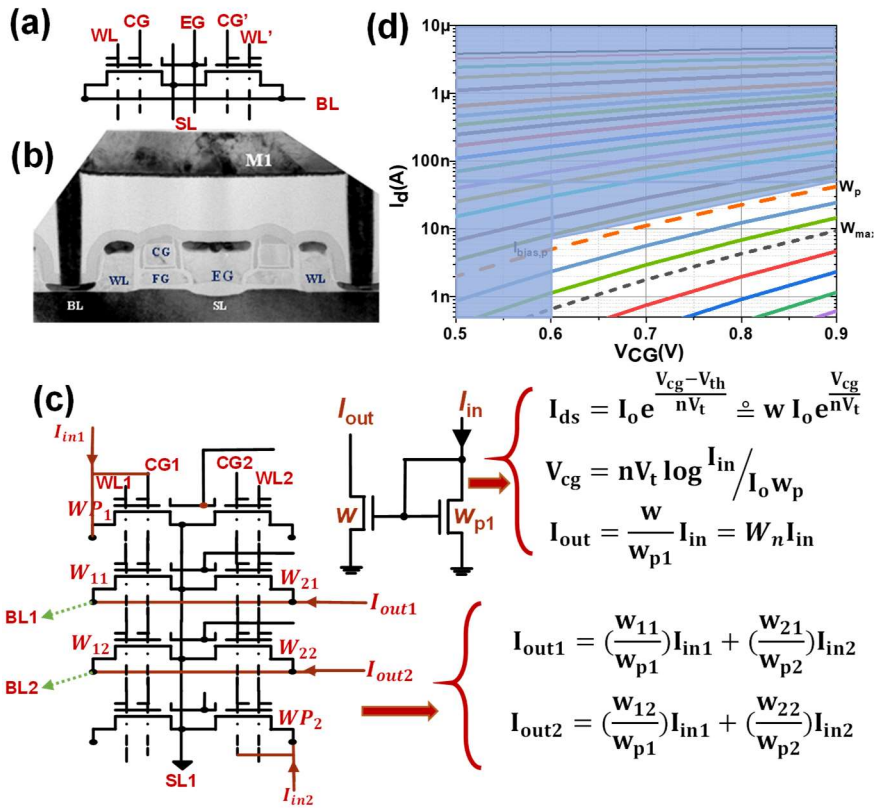


Fig. 28: Current-mode VMM implementation with split-gate NOR flash memory: (a) Schematics and (b) TEM image of SST's ESF3 supercell; (c) Drain-source current as a function of control-gate voltage under typical read conditions (V_{BL} = 1 V, V_{WL} = 1.2 V, V_{SL} = 0 V) for various programmed states. The unshaded area shows the typical low-voltage operating region; (d) Example of a 2×2 VMM circuit, including two rows of peripheral cells, and the key equations governing its operation.

The peripheral sensing circuit is typically designed to provide low input impedance on a shared bitlines (Fig. 28d), and sink/source the current flowing in it. The simplest approach for sensing the current is to use a low-voltage cascode current mirror. Its main challenges are nonlinearities in the transfer function and voltage variation on the virtual bias, which can significantly deteriorate the precision. Besides, current mirrors are susceptible to process variations, which mandates large devices. The upshot is low-speed and high-power consumption. Conventional transimpedance amplifiers (TIAs) have been used in both nanodevice computing engines and flash-based dot-product circuits to pin the virtual bias needed for linear operation and for *I-V* conversion. The area overhead of operational

amplifiers has been disregarded in favor of excellent linearity. In addition, the amplifiers are often designed to work in a certain "operating point" rather than dealing with a large-signal input. This requires a huge overdesign cost in terms of power and area for proper functionality. There are other drawbacks, including the requirement of a high gain amplifier in a TIA, the dependence of bandwidth on feedback resistor, the need for compensation, and the circuit slews for a significant period.

All-analog current-mode designs could potentially allow for a much better performance/cost. Indeed, another implementation approach is to use a second-generation current conveyor (CCII). The idea was originally introduced in [131], where CCII has been used to build current summers featuring low input impedance. Since then, various CMOS implementations were proposed [132], utilizing either open-loop and close-loop structures, with the former preferred for a better speed and dynamic behavior. It is also worth mentioning that CCII designs based on the topology introduced in [133] do not suffer from the slew-limited transient response and the gain-bandwidth product tradeoff observed in TIAs and hence, in principle, achieve higher speed compared to TIAs. However, their overall energy consumption and circuit area are still very high (see, e.g., [134]). Also, the are few designs based on operational amplifiers that are not appealing in our applications for obvious reasons.

In light of the shortcomings above, we have designed a compact current-mode peripheral circuitry based on Gilbert translinear loop, which provides a relatively low-input impedance and a wide range of gain control and temperature insensitivity. The unique design features ultimately enable excellent linearity, high-speed, and low-energy operation.

2.5.1. Proposed Circuit

The proposed circuit is shown in Fig. 29. The corresponding bitline that we intend to sense its current is connected to node "Q", and the current is supplied by M_{3a}. Due to the local feedback loop, the increase in the input current leads to a decrease in I_{2a}. This results in a differential voltage between nodes X and Y, converted to current by the subsequent low-gain amplifier. M_1, M_2, and M_4 pairs are designed in weak inversion, and M_3 pair is velocity saturated. The rest of the devices are operated in the saturation regime. When biased in weak inversion, $M_{1,4}$ pairs form a translinear loop, which has an excellent wideband current-following behavior. When the input current is zero, i.e., $I_{in} = 0$, I_{3a}=I_{3b}, the symmetrical structure of the circuit imposes $I_{3a} = I_b/2$, where I_b is the bias current provided by M_8. Since I_{in} is supplied by M_{3a}, the circuit analysis yields $I_{1a} = (I_b - I_{in})/2$ and $I_{1b} = (I_b + I_{in})/2$. Since $M_{1,4}$ pairs are biased in subthreshold, V_{XY} is expressed as $V_{XY} = nV_T \ln((I_b + I_{in})/(I_b - I_{in}))$, where V_T and n are thermal voltage and subthreshold slope factor, respectively. Furthermore, a simple analysis shows that $(I_{4a}/I_{4b}) = (I_b + I_{in})/(I_b - I_{in})$. $I_F = I_{4a}+I_{4b}$ is the bias current provided by M_9. Hence, the output current, i.e., the sensing circuit transfer characteristic, is given by $I_{out} = (I_F/I_b)I_{in}$. To improve the performance, we can use low-V_{th} devices for $M_{1,3,4}$ pairs (though this is not mandatory for proper functionality). In a 55 nm process, this allows reaching the same nonlinearity performance with crudely 15% less power consumption.

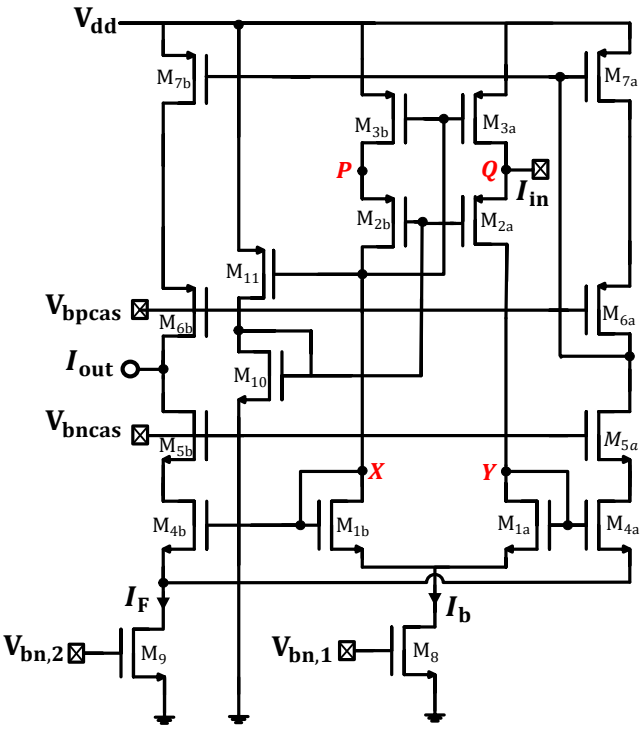


Fig. 29: The proposed sensing circuit.

2.5.2. Circuit Analysis

Closed-loop high-gain amplifiers provide excellent linearity as long as the gain requirements are met. For current processing circuits, nonlinearity becomes challenging in part due to the short channel effects in sub-deca-nm technologies. Both deterministic and random factors result in deviation from the ideal behavior of the transfer function. Specifically, the main intrinsic nonlinearity originates from unequal source-drain voltages across M_{3a} and M_{3b}. The maximum relative error, defined as $(\delta_r)_{max} = (|I_{out}-I_{out}^{ideal}|/ I_{out}^{ideal})_{max}$ due to only this factor, is shown in Figure 30a. Reducing $(\delta_r)_{max}$ is related to minimizing $\delta = I_{3a}/I_{3b}$, which in turn, is a function of I_{in} and I_b, and is achieved by designing M_3 in the deep velocity saturated region. For example, $(\delta_r)_{max}$ could be made as low as 0.1% by properly adjusting the bias current. The second issue arises from process-induced variations. A mismatch between I_{3a} and I_{3b} creates an offset in the transfer characteristics. One

straightforward solution is to adjust the conductance of memory cells accordingly. Indeed, I_{3a} - I_{3b} offset can be compensated by properly tuning conductances in two additional auxiliary columns of memory cells, i.e., with two extra devices per bitline. After measuring the input-referred offset, one of the devices of a pair, based on the sign of the offset current, is set to either sink or source the desired current while the other one is fully turned off. This approach allows avoiding scaling transistors in the sensing circuit, with minimal power/area overhead. Process-induced variations also impact $(\delta_r)_{max}$, since δ depends on the matching of the M_3 pair. Additionally, a mismatch in the voltage threshold of $M_{1,4}$ pairs could result in deviations from the ideal output current. The solution here again is to compensate total resultant offset by fine-tuning the memory devices. To evaluate the impact of process variations, we perform statistical Monte Carlo simulations of the designed circuit in a 55 nm process. As shown in Fig. 30b, both mean and variance of the total nonlinearity error could be lowered as low as 0.26%. This can be improved even further by increasing the area of the circuit (discussed below). It is worth mentioning that the discussed techniques raise energy consumption, naturally yielding a precision-energy tradeoff. Also, in practice, the nonlinearity error is expected to be less, by a factor of ~5 according to our estimates, when accounting for symmetric layout mismatch reduction techniques, which are not considered in the simulations.

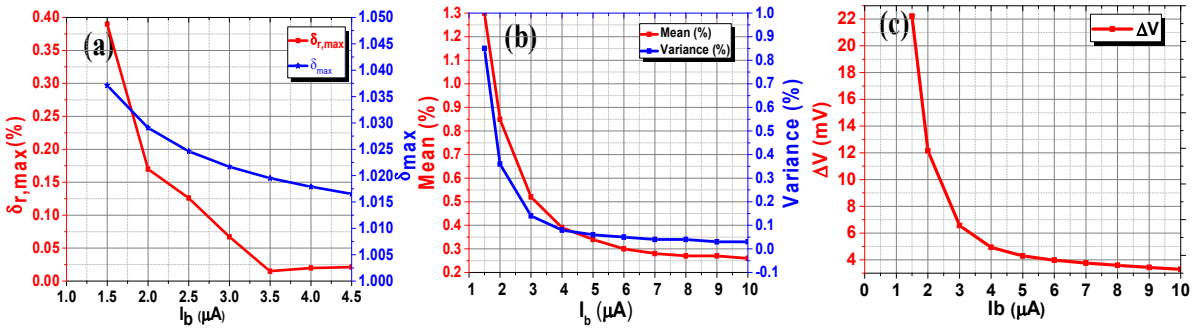


Fig. 30: The impact of nonideal transistor behavior, process variations, and finite conductance on the circuit linearity as a function of bias current: (a) Error due to the source voltage variations across M_{3a} and M_{3b} assuming $I_{in} = (I_{in})_{max}$ in TT corner, (b) total relative error at

the output due to device mismatches, and (c) virtual bias variations. For all panels, $(I_{in})_{max}$ = 1 μA and V_{DD} = 1.2 V.

Finite input conductance of the sensing circuit contributes to the nonlinearity of VMM operation rather than sensing. Intuitively, when the input current increases, I_{2a} decreases, and so does the source voltage of M_{2a}. The maximum change in the source-gate voltage of M_{2a} is given by $-nV_T \ln[1-(I_{in}/I_b)_{max}]$. However, the negative local feedback, formed by M_{10} and M_{11}, decreases the gate voltage of M_{2a} and compensates for the drift of the source-gate voltage. Additionally, proper sizing of M_{11} and controlling the bias current allow controlling virtual bias swing (or the input impedance of the circuit) for a given maximum input current (Fig. 30c). The impact of this swing on the computing precision of VMM depends on the type of is discussed later. Finally, it is noteworthy that all nonlinearity terms reduce simultaneously with respect to the bias current (Fig. 30). Therefore, in a typical design, the minimum bias current could be determined by the precision requirements.

The proposed circuit has a relatively low input-referred current noise, which scales linearly with the bias current (Fig. 31a). Hence, the linearity requirements determine the transistor sizing and, in particular, the smallest I_b and C_X, and C_Y. With these values fixed, the settling time and, as a result, energy consumption can be further optimized by finding the optimal output pole location. The output pole can be relocated by adjusting the output current, e.g., by changing I_F. For a specific translinear loop size determined by the linearity requirements, increasing I_F initially improves the settling time (Fig. 31b). However, at some point, the overshoot in time response becomes excessive and deteriorates the settling time. Increasing the output current is no longer helpful since the dominant pole is no longer attributed to the output pole. The optimum settling time is obtained by adjusting I_F based on given I_b, C_X, and C_Y, i.e., the location of the first pole, and C_L, i.e., corresponding dimensions

of the loaded array or the next stage. Fig. 31c shows the temperature dependence of the considered nonlinearities. In general, virtual bias is sensitive to temperature variations because V_t is a function of temperature. To reduce the worst-case ΔV below the desired value across the entire temperature range, we may supply I_b from a PTAT (proportional to absolute temperature) current source. This compensation scheme limits the variations in the virtual bias within a wide range of temperatures, as shown in Fig. 31c. Note that I_F is also supplied by the same PTAT source to keep the slope of the transfer function temperature invariant (within < 0.2%). The temperature sensitivity of both ΔV and the slope of the transfer function can be further improved by designing a more complex compensation circuitry. Finally, simulation results indicate that reasonable ±4% fluctuations in supply voltage result in <0.5% change of the transfer function slope (Fig. 31d), which stems from the fact that the slope depends only on bias currents. Hence, as long as the current reference that supplies these bias currents is voltage insensitive and critical devices remain in their targeted operating region, the linearity remains acceptable.

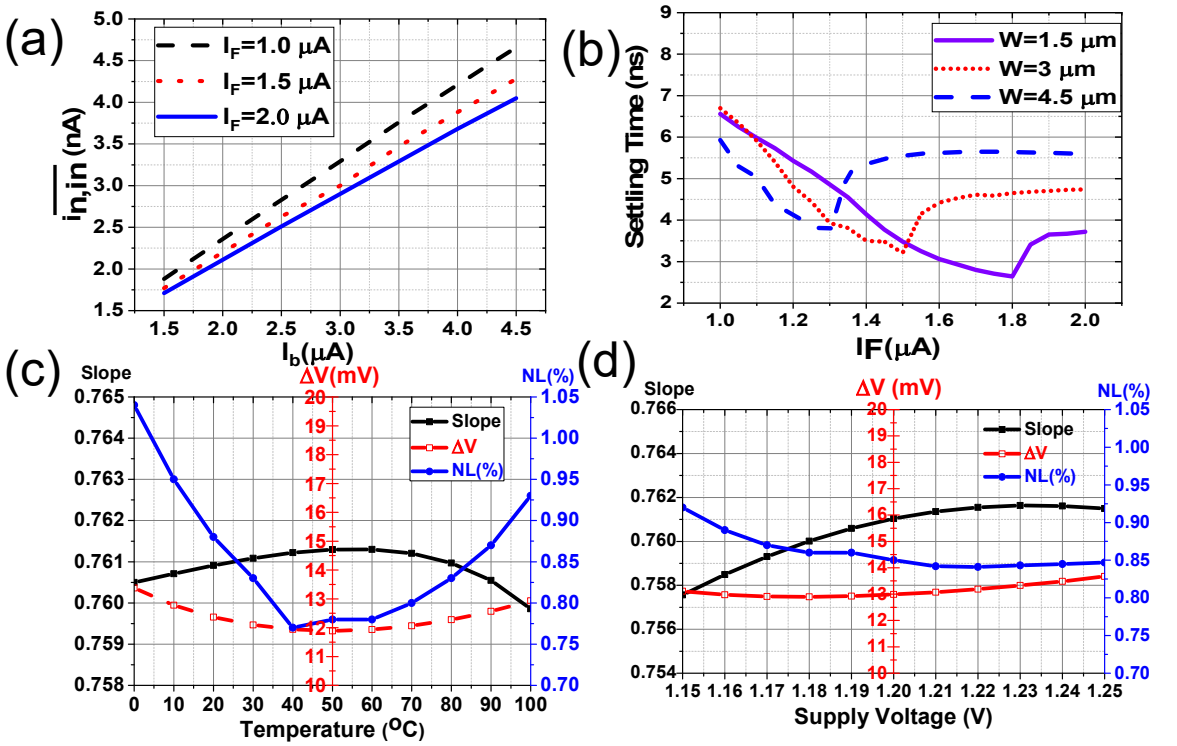


Fig. 31: Analysis of noise, settling time, and PVT variations: (a) Total integrated input-referred current noise for several I_F as a function of bias currents at I_{in} = 0 and 100 MHz bandwidth; (b) Settling time as a function of translinear loop size and I_F current at I_b = 1.75 µA; (c) Temperature dependence of virtual bias variation (ΔV), slope, and total worst-case nonlinearity error; (d) Impact of supply voltage variations on ΔV, transfer function slope, and total worst-case nonlinearity error. For all panels, $(I_{in})_{max}$ = 1 µA, C_L = 7.5 fF.

2.5.3. VMM Design Case Studies

We designed four different styles of the proposed sensing circuit, in each case optimizing bias current and size of the devices according to the specific metric. In particular, we consider the power-optimal (referred to as S_1), the area-optimal (S_2), the precision-optimal (S_3), and the energy-optimal (S_4) designs. Besides, each style is implemented based on a targeted 6-bit for $S_{1,2,4}$ and 8-bit for S_3 precision requirement. All designs are based on 1.2 V devices in Global Foundries 55 nm process technology. Fig. 32a summarizes various characteristics of the implemented designs, while Fig. 32b shows the impact of input current for S_4. (The maximum input current is naturally linearly proportional to the size of the VMM circuit.) Figure 6b shows that for small input currents, the critical devices must be kept large to

counteract the process variation effects, resulting in slower operation. On the other hand, I_b and width of $M_{1,2,3}$ can be scaled up accordingly for larger maximum input currents to keep the precision/speed constant. The power, area, and energy naturally increase with respect to the maximum input current. The impact of process variations on the circuit's linearity is also studied for all designs (Fig. 32a). Statistical simulations across all corners show that the sensing circuitry can effectively operate with up to 8-bit precision. In particular, the process-induced precision errors are controlled by the proper sizing of the translinear devices and the bias current. Dispersion in transfer function slope, which follows a normal distribution, is addressed by adjusting the weights.

Energy-optimal design S_4 is further utilized to investigate the performance of a current-input current-output fully analog VMM based on split-gate embedded NOR flash memory technology. One of the most important characteristics of the analog-mode VMM is its effective operating precision. Even though the S_4 design is suitable for 6-bit operation, for simplicity, we here consider a rather conservative assumption that VMM's input, weight, and computing precisions are all effectively 4 bits. Note that the weight precision might be limited by each of the following factors: tuning accuracy, drift, bitline bias variations, and subthreshold slope nonlinearities, depending on the properties of the technology. The redesigned layout of the memory array allowed to demonstrate experimentally >6-bit tuning accuracy for a single cell when a sufficient number of pulses are applied during the tuning procedure, which is far better than the targeted 4-bit precision. Accelerated retention tests show negligible drift after many years of operation, and the virtual bias variation ΔV can be limited to less than 15 mV, which corresponds to < 1% overall bitline distortion for the targeted current range. In general, the effective weight error due to the subthreshold slope

nonlinearities depends on the choice of peripheral device state and the selected range of states used for the array devices. In addition, there is a tradeoff between power consumption and weight precision. Indeed, using the memory states corresponding to the lower operating voltages (Fig. 28c) reduces the power consumption. The downside is that the subthreshold slopes are more nonlinear in those regimes. In light of this tradeoff, the state of the peripheral cell and the maximum current via array device are assumed to be 30 nA and 10 nA, respectively, under V_{CG} = 0.9 V, V_{WL} = 1.2 V, and V_{BL} = 1 V biasing conditions, shown in the unshaded region of Fig. 28c. Since the input-referred noise of the sensing circuit is negligible, the main limiting factors for the computing (output) precision are the nonlinearity of the sensing circuit and the noise of the memory devices.

The simulation results in Fig. 32 show that the S_4 design style offers a total relative nonlinearity error of ~1.1%. The subthreshold current fluctuations are mainly due to random telegraph noise (RTN) in memory cells and, more generally, 1/*f* noise after repeated switching. Our measurements show that only a few ESF3 cells (out of 140 total) had considerable subthreshold current fluctuations, even after cycling each device 1000 times. The root mean square of the current noise via a single device operating in the maximum target current is ~575 pA at 300 MHz operating bandwidth (500 kHz corner frequency). The total resultant output signal-to-noise ratio is ~44.8 dB for a 100×100 dot-product operation. (Due to similar physics of operation, 1/*f* noise can also be crudely quantified by considering much more numerous reported noise data for standard 55 nm MOSFETs with the same width and length.) The above analysis takes into account all important nonideality factors and shows that achieving 4-bit computing and weight precision should be relatively straightforward for the considered VMM

design. The estimates are rather conservative and, e.g., even higher weight precision is possible when using larger operating voltages.

To evaluate and optimize operation speed, we assumed that several VMMs are chained in a cascade structure, with the output of one VMM sensing circuit feeding the input of the next VMM stage directly. This assumption represents an all-analog multilayer neural network implementation (though it neglects additional circuitry, which might be required for neuron implementation). The propagation delay through such cascade can be minimized by adjusting the output poles of VMMs. Note that the input pole of a particular stage VMM is effectively the output pole for the preceding stage multiplier. Also, out of the two poles in each sensing circuit, the first one is always fixed and determined by the targeted maximum input current and linearity requirements; however, the second one can be optimized for minimizing the total settling time of the circuit. More specifically, for a desired, fixed sensing circuit linearity and given capacitive load, the smallest delay is achieved at the specific sensing circuit output current. However, the optimal current value is typically higher than the nominal subthreshold current of the minimum size floating gate transistor. Forcing such optimal current via a single peripheral floating gate cell would significantly reduce errors in the multiplier operation. To overcome this issue, we assume that M_p peripheral cells are connected in parallel for each input, which effectively increases the width of the peripheral floating-gate transistors. In particular, let us first note that the optimal output current is proportional to the load capacitance, which is $(M+M_p)C_{cell}$, where C_{cell} is the unit capacitance of the devices. Therefore, for the most interesting cases of large M, increasing M_p and, simultaneously, output current for optimal pole location will lower individual currents via peripheral cells and decrease settling time. More generally, the settling time, in this case, is proportional to

$(1+M/M_p)C_{cell}/(I_p)_{max}$, where $(I_p)_{max}$ is the desired maximum current to which the peripheral cells are tuned. Fig. 32c summarizes various performance characteristics of the considered VMM as a function of its size. As expected, the simulation results show that the average energy consumption for the dot-product operation (one channel) is growing superlinearly with N, mostly due to the increased maximum input current. The number of operations per channel grows linearly, and with constant settling time, the energy-efficiency saturates. The relative peripheral area overhead is always below 11%.

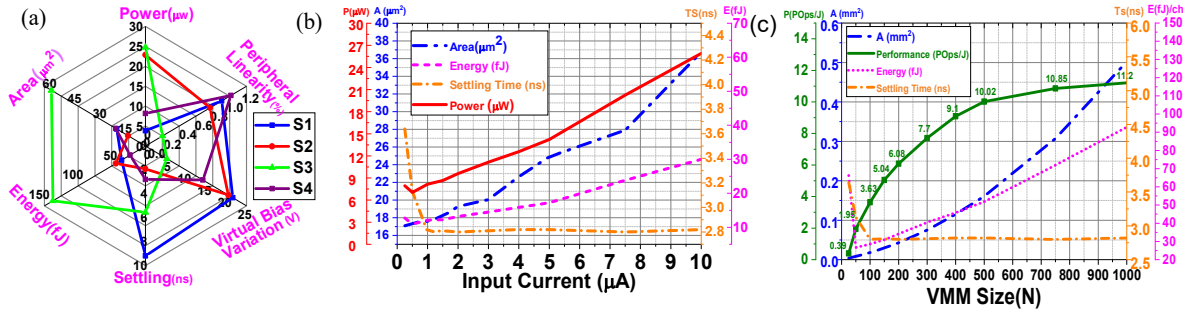


Fig. 32:Analysis of noise, settling time, and PVT variations and VMM-level performance: (a) Total integrated input-referred current noise for several I_F as a function of bias currents at I_{in} = 0 and 100 MHz bandwidth; (b) Settling time as a function of translinear loop size and I_F current at I_b = 1.75 µA; (c) Temperature dependence of virtual bias variation (ΔV), slope, and total worst-case nonlinearity error; (d) Impact of supply voltage variations on ΔV, transfer function slope, and total worst-case nonlinearity error. For all panels, $(I_{in})_{max}$ = 1 µA, C_L = 7.5 fF. (c) Performace, total area, and energy per channel for $N \times N$ VMM based on 55 nm ESF3 NOR flash memory. POp/J operation is achieved for $N > 50$, which are practical kernel sizes for many applications.

In summary, a very efficient sensing circuitry that utilizes the translinear principle of Gilbert cell is proposed to boost the performance of NVM-based analog-mode VMMs. In prior work, the area, energy, and density potentials of current-domain circuits are typically counterbalanced by the overhead of PVT compensation. Here, offset calibration is performed by considering two extra columns of programmable NVMs in the crossbar array so that robustness against PVT variations is achieved with minimal overhead. Our simulation results

show that 100×100 4-bit VMM designed in 55 nm CMOS technology with embedded NOR flash and employing energy-optimal sensing circuit achieves 3.63 POps/J.

2.6. Low-Power Mixed-Signal VMM Design

An internally analog, externally digital VMM circuit offers the best of both worlds: The density and energy efficiency of the analog domain and the noise-robustness and versatility of digital communication. Accordingly, mixed-signal VMMs have been realized in a variety of applications, including neural networks, support vector machines, and IoT systems. In this part, an energy-efficient analog mode VMM circuit with a digital input/output interface and configurable precision is proposed. Similar to some previous work, the computation is performed by a gate-coupled circuit utilizing embedded floating gate memories. We use the ultra-low-power sensing circuitry that we designed in the previous section to implement the local sensing circuit. Additionally, the digital-to-analog input conversion is merged with VMM, while the current-mode algorithmic analog-to-digital circuit is employed at the circuit backend. Such implementations of conversion and sensing allow for circuit operation entirely in the current domain, resulting in high performance and energy efficiency. Post-layout simulation results for 400×400 5-bit VMM circuit designed in 55 nm process with embedded NOR flash memory show up to 400 MHz operation, 1.68 POps/J energy efficiency, and 39.45 TOps/mm^2 computing throughput.

Time-based VMMs [135] and switch-capacitor multipliers [136] use charge to encode data. The former approach, designed to operate in very low voltages, is based on charge integration from digitally programmable current sources. One of the challenges is process-voltage-temperature variations that may limit the smallest integration delay and hence the circuit performance. For the latter case, metal fringing capacitors have been exploited to build

VMM circuits with moderate computing precision. These topologies have been explored for implementing (> 4 bit) multipliers using bulky and power-hungry active amplifiers. In the passive version of such circuits, amplifiers are eliminated [137], which can lead to potentially more power-efficient and faster design. The main challenges, however, are leakage, capacitive coupling, and charge injection issues, which confine passive switch-capacitor approaches to 2-3 bit resolutions. In another approach, current/voltage is employed as a state variable. For example, a VMM circuit with digitally controllable single MOS-based current sources, in which the width of the transistors are scaled according to the predetermined weights, has demonstrated very high energy efficiency [138]. The main caveat of such design is an area (and hence energy) overhead for weight implementation, which exponentially increases with weight precision.

2.6.1. Circuit Architecture

A digital-input digital-output (DIDO) $M\times N$ VMM circuit computes M P_{o}-bit dot-products between N-element P_{in}-bit input vector and corresponding N-element vector of P_{w}-bit weights in parallel. Note that, in general, the precision of dot-product computation might be higher compared to that of ADC converter. Specifically, the top-level architecture of the proposed circuit is shown in Fig. 33. In this architecture, data are buffered into a shift register to hold it during the processing, which is triggered by φ_1 control signal. Upon completing the data transfer, digital voltages are applied to the array to generate currents in each channel proportional to the dot-product of input and weight vectors. A merged-DAC (MDAC) architecture is employed at the input interface to reduce conversion overhead. In this case, each matrix weight W_{ji} is implemented with a set of P_{in} FG devices, i.e. $W_{ji}^{k} = 2^{k}/(2^{P_{\mathrm{in}}}-1)\,W_{ji}$, $P\mathrm{in} \geq k \geq 1$, where k is the input bit significance. Assume that i^{th} input

is a binary vector $\{b_{P_{\text{in}}}, \ldots, b_1\}$, in that case, a current injected by the memory cells implementing weight W_{ji} to the j^{th} output is given by: $I_{ji} = \sum_{k=1}^{P_{\text{in}}} b_k \, 2^k/(2^{P_{\text{in}}} - 1) \, W_{ji}$. Negative weights with FG memory devices are implemented using differential pair of weights $W_{ji} = W_{ji}^{+} - W_{ji}^{-}$. Hence, for the two-quadrant VMM implementation, the total current in the j^{th} differential output is given by $I_j = \sum_{i=1}^{N} \left(I_{ji}^{+} - I_{ji}^{-} \right)$. The proposed MDAC implementation is based on the analog synapses, which are implementing the weights as well. Therefore, MDAC's area and energy are contributed by the additional $2 \times M \times (P_{\text{in}}\text{-}1)$ array of FG cells.

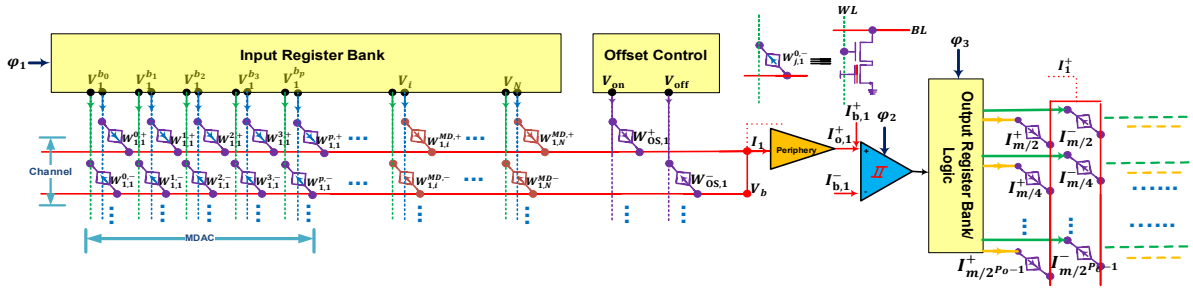


Fig. 33: One channel of the proposed two-quadrant VMM circuit with digital inputs and outputs. Here, we assume that inputs and outputs are non-negative, while weights can be negative or positive.

2.6.2. *Floating-Gate Memory Array and Sensing Circuit*

FG memory array is implemented using split-gate ESF3 SuperFlash®, which is a commercialized embedded NOR flash technology developed by SST Inc. The ESF3 flash memory is very desirable for the realization of couple-gate arrays, as illustrated in chapter 2.2. The robust subthreshold operation of ESF3 devices is typically in the range of 100 pA - 300 nA range. In our design, the least significant bit of the weights is 500 pA. Maximum achievable P_{w} depends on the state drift, tuning accuracy, and virtual bias variations. In order to have 5-bit effective precision, we bound the weight error by 0.9%, i.e., the normalized difference between the desired and actual subthreshold currents of the cell should be less than 0.9%. We may achieve this condition by ensuring the tuning accuracy (which could be

improved with increasing write time) and drift are bounded by 0.4% [40], and hence the error due to maximum sustainable bitline voltage distortion can be < 0.5%. The virtual bias variations impact the absolute value of the weight via channel length modulation and drain-induced barrier lowering. For the utilized range, 0.5% crudely translates into ΔV_b = 10 mV. The sensing circuit provides the virtual voltage V_b on shared bit (i.e., the horizontal lines on Fig. 33) lines. The design of the sensing circuit has been described in section 2.2. The proper sizing of M_{11} (Fig. 29) and adjustment of bias current I_b allows reducing ΔV_b to 3 mV, which ensures 5-bit weight precision.

2.6.3. Current-Mode ADC Design

Algorithmic ADCs feature high resolution, throughput, and small area. Among such architectures, conventional current-mode ADCs typically offer the best speed-area performance [139]. In our work, we use current-mode cyclic ADC to minimize the conversion cost and, more importantly, to leverage the tunability of FG cells for the precise current generation. Specifically, a 1-bit per stage cyclic current-mode ADC is implemented, as shown in Fig. 34. Note that we have not used the common 1.5-bit per stage design since it has a significant overhead. Instead, the offset of the comparator is compensated by adjusting input bias currents I_{BA} for each channel using FG cells. Although these bias currents contribute to static power consumption, they are critical to support a bipolar output and keep the mirror devices at the input stage of the comparator turned on, which facilitates faster conversion. The constant current sources are generated by auxiliary MDAC arrays of FG devices, which share bit lines with the main array. The operation is performed in a sequence of P_o steps: In the first step, the current comparator determines whether the input current (I_{in}), fed by sensing circuitry, is positive or not and generates a sign bit. In the next cycle, based on the sign bit,

$I_{max}/2$ is either subtracted from or added to input current, where I_{max} is the maximum possible amplitude of ADC input current. At the end of k-th step, the residual current is given by: $I_{res} = I_{in} + \sum_{l=1}^{k-1}(-1)^{D_{Po}-l+1}(I_{max}/2^l)$, k>1, where D_l represents l–th output bit. The process is repeated until LSB (D_1) is generated. Then, $D_{<Po:1>}$ is buffered to a parallel register. The operation needs minimum control, and the control circuitry is shared between all channels, leading to a very compact implementation. The controller is essentially simple logic circuits to enable the process and a shift register, which is cleared at the end of each conversion and shifts logic "1" at each conversion step.

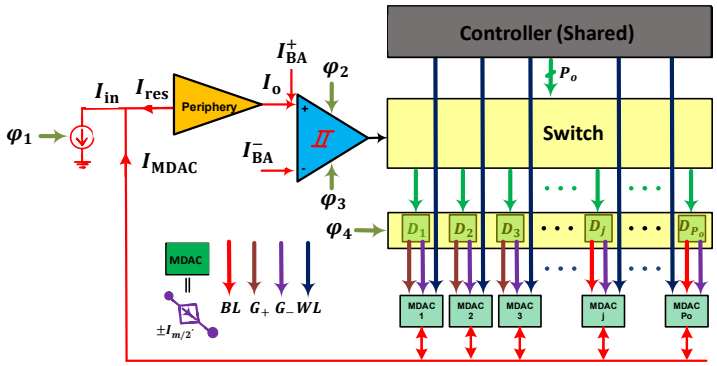


Fig. 34: Block diagram of the proposed algorithmic ADC.

The comparator design is shown in Fig. 35a. The circuit utilizes a cascode current mirror as a preamplifier and a latch stage similar to that of StrongARM [140]. At the beginning of each step, when φ_2 or φ_3 is high, nodes X, Y, P, and Q are grounded (shielding mode). The purpose of shielding is to reset the state of the comparator and avoid storing excess charge on node X right after each conversion, which may happen due to the delay of the peripheral circuit. Shielding continues until I_{6b} restores to I_b^+. In the following, both φ_2 and φ_3 go down, while the current $I_{6b} + I_{com}$, where $I_{com} = (I_{M4a} + I_{M4b})/2$, charges node X. The circuit operation is similar for V_Y, with $I_b^- + I_{com}$ charging node Y. I_{com} is used to inject a dynamic common-mode current to turn on $M_{1a,b}$ quickly. When V_X reaches the threshold of transistor M_{1a}, φ_3 goes high, and regeneration begins. Finally, cross-coupled transistors turn on, and the

differential current, amplified by the positive feedback loop, brings one of the outputs to V_{DD}. An example of this operation is shown in Fig. 35b, in which the input current is 700 nA and $(I_{in})_{max}$ = 1 µA. Since input current is positive, Q becomes "1" after comparison is finalized, and 500 nA is subtracted in the following step. The process repeats itself until an LSB is generated. Due to the unique features of analog-grade FG cells, which are exploited in performing multiplication, offset compensation, and generation of constant scaled current sources, the sizing of the proposed circuit is relaxed. Indeed, the high performance achieved in the proposed ADC topology stems from three factors: low-overhead offset compensation, which relaxes the tradeoff between speed and resolution, embedded design of current references with zero power overhead, and low-power design of dynamic current comparator. We design the comparator in a 55 nm CMOS process that settles at 0.65 ns for 30 nA differential input current while dissipating only 2.07 µW dynamic power on average. At high precisions, the clocking scheme of cyclic ADC can be redesigned to improve its energy efficiency . In addition, the input-referred current noise of the comparator (and the sensing circuit) is much smaller than the noise associated with FG cells.

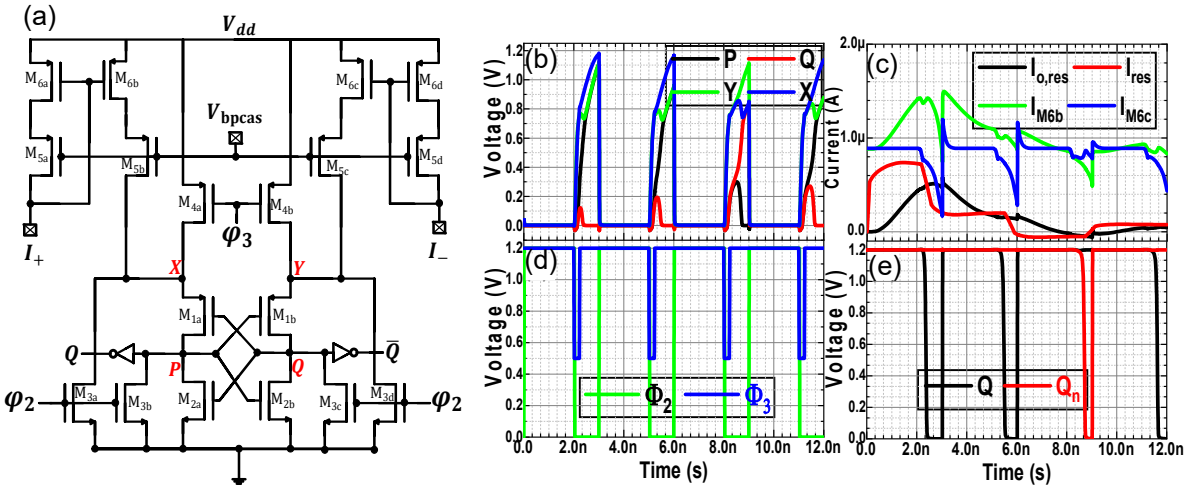


Fig. 35: (a) The dynamic current comparator circuit. (b-e) The timing diagram of the ADC (shown for four cycles): (b) transient voltage of nodes X, Y, P, and Q, (c) transient residual

currents and drain currents of M6c,6b, (d) clocking scheme, and (e) corresponding digital outputs.

2.6.4. DIDO VMM Simulation Results

The proposed DIDO VMM circuit is simulated in GlobalFoundries 55-nm LPe 2P8M process technology. The design is optimized to maximize energy efficiency. The dynamic power (in comparator and array) and the static power (in peripheral circuitry and array) are both included in power consumption estimates. The same precision for inputs and weights, which is limited to ~ 5-bit as discussed above, is assumed. When the offset of the sensing circuit and comparator are properly compensated, the output precision becomes bounded by the device noise, which can be controlled by adjusting the bandwidth or current ranges of the devices. Our estimates show that >8-bit precision on the full-scale current range is feasible for $N > 25$. The simulation results provided in Fig. 36a indicate the circuit area grows rapidly as a function the precision, mainly due to the merged-DAC overhead. The same trend is observed in the settling time and energy due to the cyclic structure of the ADC. The trends for delay and energy consumption at low precisions (e.g., at < 3-bit) are explained through the argument that the sensing circuit cannot be scaled down to preserve the tolerance to process variations. Throughput (TH) decreases as expected because the same number of operations are performed slower. For the same reasons, energy efficiency (EE) and area efficiency (AE) gradually decrease as precision increases. On the other hand, with input/weight precision fixed at 5-bits, the total active area does not change much with output precision since ADCs have a small area overhead. The number of operations grows quadratically as a function of VMM size, and so does the total active area (Fig. 36c). At low currents (smaller size VMMs), the sensing circuit is slower. Because of that, TH is increasing roughly quadratically with VMM size. Though the total energy consumption is increasing with VMM size, the EE is also increasing

because of TH and is saturating at ~1.8 POps/J for $N > 500$. Fig. 36d shows energy breakdown for several VMM circuit implementations. Peripheral circuitry and ADCs are typically the major sources of energy consumption. The power consumption of the proposed floating-gate-based ADC is ~6 μW per channel and almost the same for all designs. For the first, a relatively small VMM circuit is designed at 4 bits, and hence the array and sensing power are less than the power consumed in the comparator. For larger precisions and VMM circuits, i.e., the second and third considered cases, respectively, the contribution of sensing circuitry becomes more prominent. The area breakdown is also provided in Fig. 36e and shows that with the optimal design of the peripheries, the FG array dominates the area for larger VMM circuits.

The performance metrics of our design compare favorably with the best-reported results. For example, Ref. [125] reports a memristor-based dot-product engine with an estimated 30 TOps/J maximum energy efficiency for a 128×128 crossbar circuit. State-of-the-art low-precision switch-capacitor VMM circuits report a serial 6b/3b/6b VMM circuit in 40 nm CMOS [136] that achieves 7.72 TOps/J in 0.012 mm^2, and a 9.61 TOps/J in ~0.011 mm^2 for an 8b/14b/8b 16-parallel channels in 28 nm CMOS [141]. For comparison, the proposed approach achieves 1.68 POps/J for a 400×400 VMM circuit (in simulations) when computation, I/O, and weights are all at 5-bit precision, which is ~100× better than that of state-of-the-art switch-capacitor design. In principle, for both switch-capacitor designs, weights can be programmed quickly, making it suitable for a larger range of applications, as compared with the proposed design. However, this advantage often comes with the cost of bandwidth limitations and significant overhead for moving weights in large-scale systems. FG memory-based circuits [100] are also used to realize a neural network featuring 7.2-bit weight

precision in a 130 nm CMOS process. Though the system is fully analog, it achieves ~1 TOps/J while occupying 0.36 mm^2 [130].

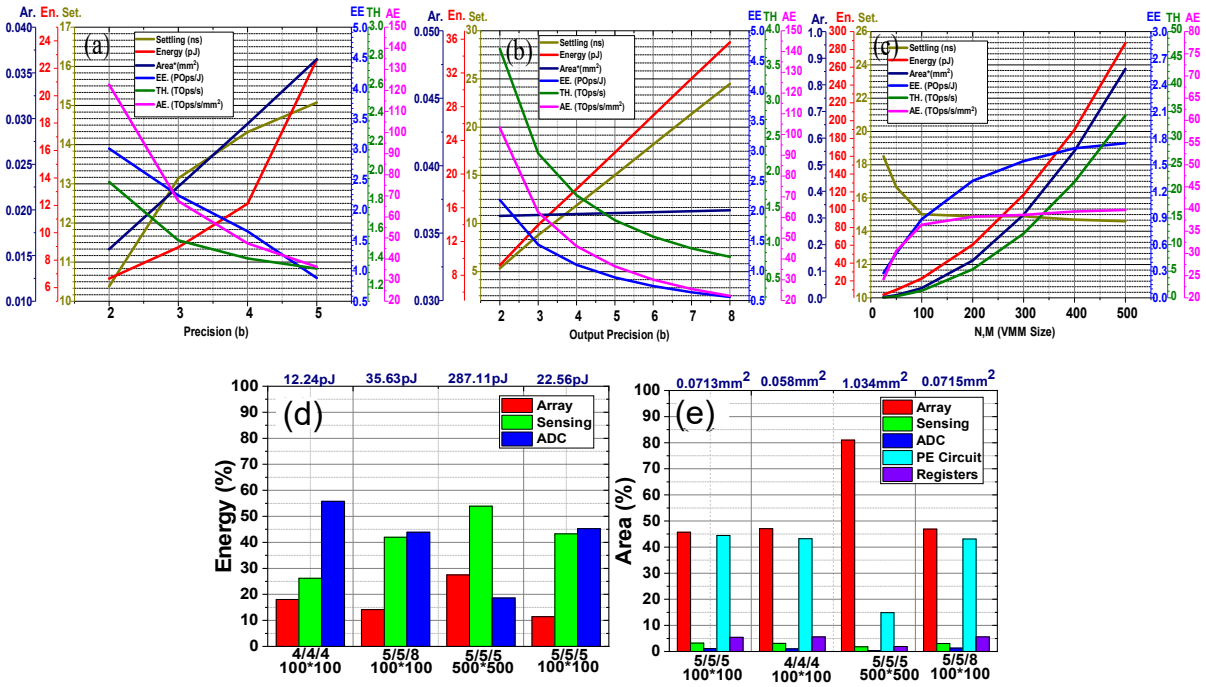


Fig. 36: Various performance metrics of DIDO VMM circuit as a function of (a) precision assuming P_i=P_w=P_o and M=N=100, (b) output precision P_o, assuming P_i=P_w=5 and M=N=100, and (c) VMM dimensions, assuming P_i=P_w=P_o=5. (a) Energy and (b) area breakdown of 4 different DIDO VMM implementations.

2.7. Designing DNN Inference Accelerators

This section describes the architecture and properties of a mixed-signal convolutional neural network (ConvNet) chip based on embedded floating-gate cell arrays redesigned from a commercial 55 nm NOR flash memory, which can be used to perform image classification on CIFAR-10, CIFAR-100, or MNIST datasets. Each array performs high-speed and energy-efficient vector-by-matrix multiplication in the analog domain. The network-specific neuromorphic chip is designed and fabricated in the Global Foundries 55 nm LPe CMOS process and includes ~13.4×10^6 total synapses, ~1.5×10^6 training parameters, ~3×10^3 neurons while occupying ~43 mm^2. Theoretical simulations indicate the chip achieves ~87.5%

accuracy on CIFAR-10 while consuming, ~2.1 µs/pattern latency, and ~3.37 µJ/pattern and offers ~98% classification accuracy on MNIST while achieving a performance of ~0.036 µs/pattern latency and ~324 nJ/pattern energy.

Other unique features of this design include: time-division multiplexing (TDM) to reduce network complexity, NOR-flash-based DAC to reduce the overhead of data converters, merged implementation of convolutional and average pooling operations, local and global folded wiring to minimize the redundancy and maximize data reuse, the novel design of tuning circuits, sharing tuning circuits among eFlash arrays, programmable gain/activation function slope to improve SNR with unique test access for precise mapping, individual access to each neuron output for advanced training schemes, temperature resiliency, and more importantly, massive read/write/erase parallelism, which enable sub-minute high precision tuning of the entire chip. The fabricated neuromorphic circuit is trained ex-situ; that is, the model weights are initially computed on a precursor server in the training phase. The weights are converted to state currents of flash synapses in the programming mode. Finally, the hardware is operated in the inference phase to perform the inference task and image classification.

Fig. 37a shows the architecture of the implemented ConvNet, which classifies 32×32 4-bit RGB (or 64×64 3-bit) images and includes 2 convolutional, 2 average pooling, and 2 fully-connected (FC) layers. The convolutional operations in the first two layers are performed fully in the analog domain in 25 time-multiplexed steps (Fig. 37b). In every step, one pixel per map in L2 is generated. The outputs of the second layer are digitized using a bank of 120 4-bit ADCs and then stored in register banks, which drive the first fully-connected layer. The two FC layers operate in a single cycle to produce the 120 outputs of the network. The mixed-signal accelerator also supports an MLP-mode, which can be used to perform extremely high-

speed classification of 64×64 3-bit grayscale or 32×32 RGB 4-bit images. In MLP-mode, the input images are directly fed to the last two fully-connected layers to perform a one-shot classification.

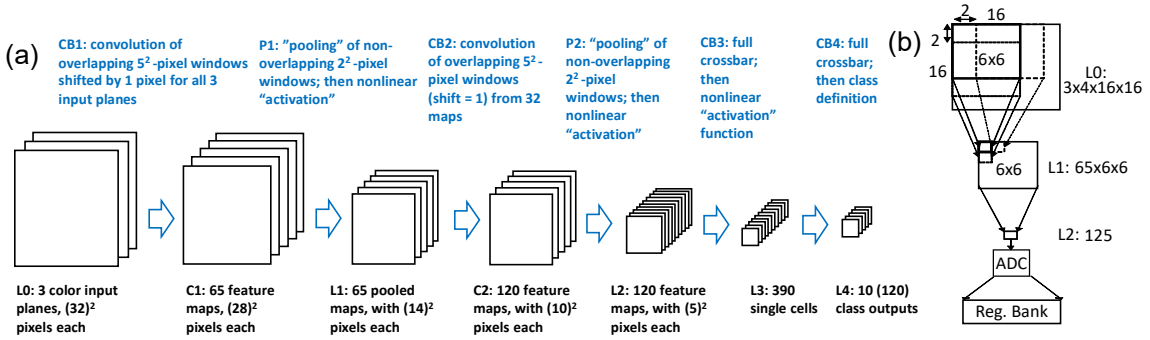


Fig. 37: Network-specific mixed-signal image classifier accelerator: (a) The architecture of the implemented deep convolutional neural network, (b) TDM operation of the chip.

The fabricated 6.5 mm × 6.7 mm chip in 55 nm process is shown in Fig. 38a, and the adapter to interface the chip with our measurement setup is provided in Fig. 38b. Future work focuses on performing measurements and characterization of the chip.

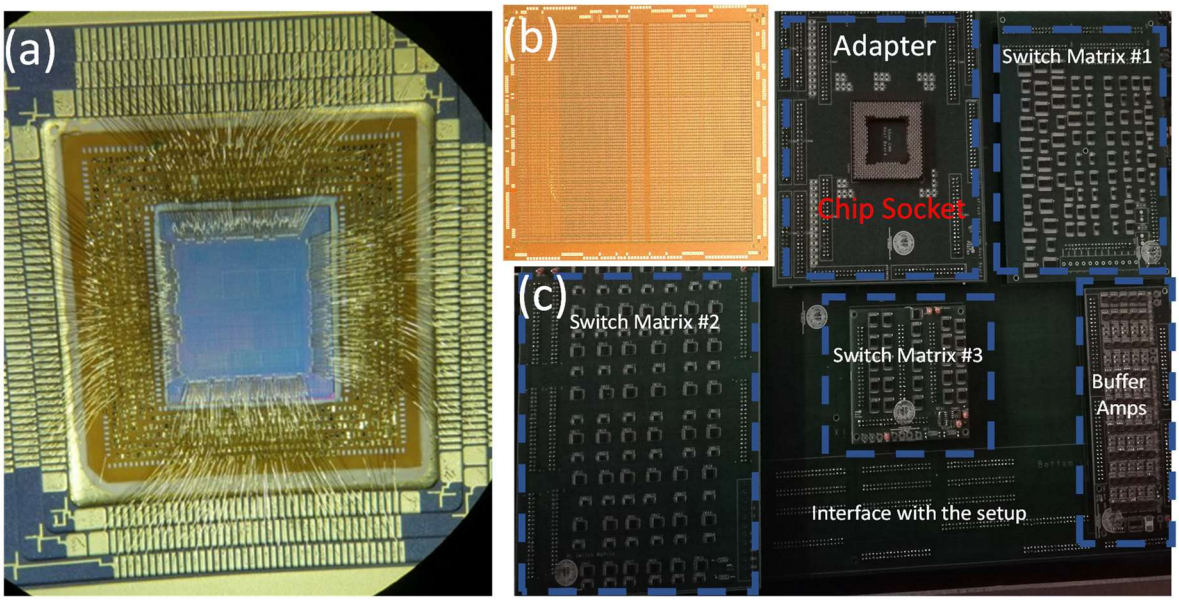


Fig. 38: (a) Packaged chip. (b) Micrograph of the chip. (c) Fabricated adapter board to interface the chip with the general-purpose setup.

We have also designed and fabricated a prototype of a 56 × 56 4-bit input, 120-output, three-layer neuromorphic network based on arrays of highly optimized redesigned eFlash

memories in GF' 55 m LPe CMOS process. The neuromorphic network is a 3136-390-120 multilayer perceptron, which includes 1.2×10^6 trainable parameters and 510 neurons. All active circuitry has a total area of <8 mm^2. Simulation results indicate that the network can classify MNIST images with ~98% accuracy, and the classification of one pattern takes ~100 ns and 0.25 nJ—4.3%, 10×, 100× better than our previous implementation [40] based on 180 nm eFlash technology. The improved performance originates not only from exploiting a more advanced process node but also from employing more optimum peripheral circuits.

Similar to the ConvNet chip, this chip is also trained ex-situ with weights computed in a precursor software and then transferred to the chip in a tuning process. We also use similar tuning circuits for each block, which allow us to tune 30 devices (5 devices in a block) in parallel for the entire network. Hence, we only focus on several key differences of this design with the previous one.

Fig. 39a shows the overall architecture of the implemented MLP, which can classify images with up to 56×56 4-bit pixels, and Fig. 39b shows the logical implementation of this design. Unlike the previous circuit that used TDM operation, this network is fully analog and operates in a single cycle. A massive eFlash array is needed to implement the 3136×390 current-input current-output dot-product in the first layer. 390 differential neurons are connected to sense the currents from the outputs of the first layer, rectify the response, and apply to the synapses in the second layer. The 390×120 current-input current-output VMM and 120 connected neurons generate the predicted response.

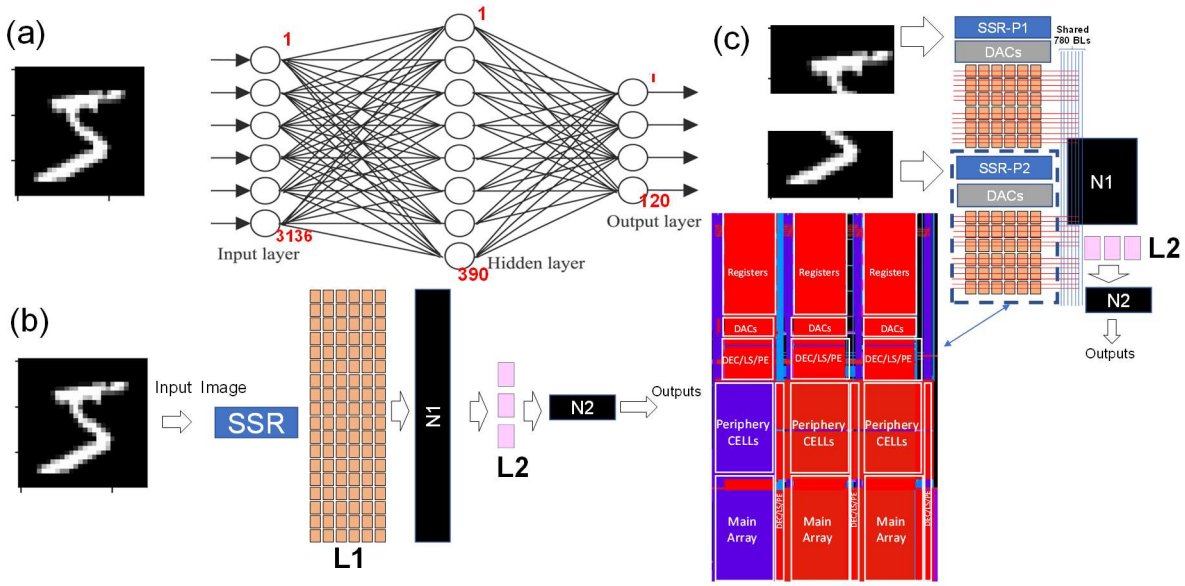


Fig. 39: (a) The network architecture. (b) The logical implementation of the image classifier (SSR: shift registers, L1: eFlash arrays implementing the first layer, N1: neurons in the first layer, L2: eFlash arrays implementing the second layer, and N2: neurons in the second layer). (c) The floorplan of the folded design with BLs shared on the side of the first layer. Panel (c) also shows the layout of a superblock from the first layer.

The chip is expected to operate reliably across the 0-100 temperature range due to the use of temperature-insensitive DACs, sensing circuits, and an optimal weight to state current mapping [120], which improves the resiliency toward temperature variations. With 4-bit input and at least 5-bit weight precision, our simulations indicate we can achieve ~98% prediction accuracy of the MNIST dataset using this network.

Fig. 40a shows the layout of the entire chip, and Fig. 40b shows the micrograph of the fabricated chip. Besides this classifier, we have also prototyped standalone VMM circuits (which were discussed in chapter 2) and characterization blocks in this tape-out: 30×30, 100×100, and 300×300 energy-efficient time-based (see section 2) differential VMMs [164] are implemented with complete peripheral circuits, including digital neurons, digital-to-time, and time-to-digital converters, and a 1 GHz phase-locked loop to generate a high-speed clock for timing circuits. 20 current-mode 256×256 blocks with merged-DAC architecture (see section 2) are also prototyped. We also have designed 20 10×10 VMM blocks in which we

have changed the various aspect of the eFlash cells, e.g., the length of the gates. In one case, we have fabricated a PMOS eFlash array to study whether the same device performance can be achieved with these complementary devices. Such feature reduces the complexity of peripheral circuits as the sinking and sourcing of the current could be done in a single node, which ultimately leads to improved energy efficiency in VMMs. Future work focuses on performing measurements and characterization of the chip.

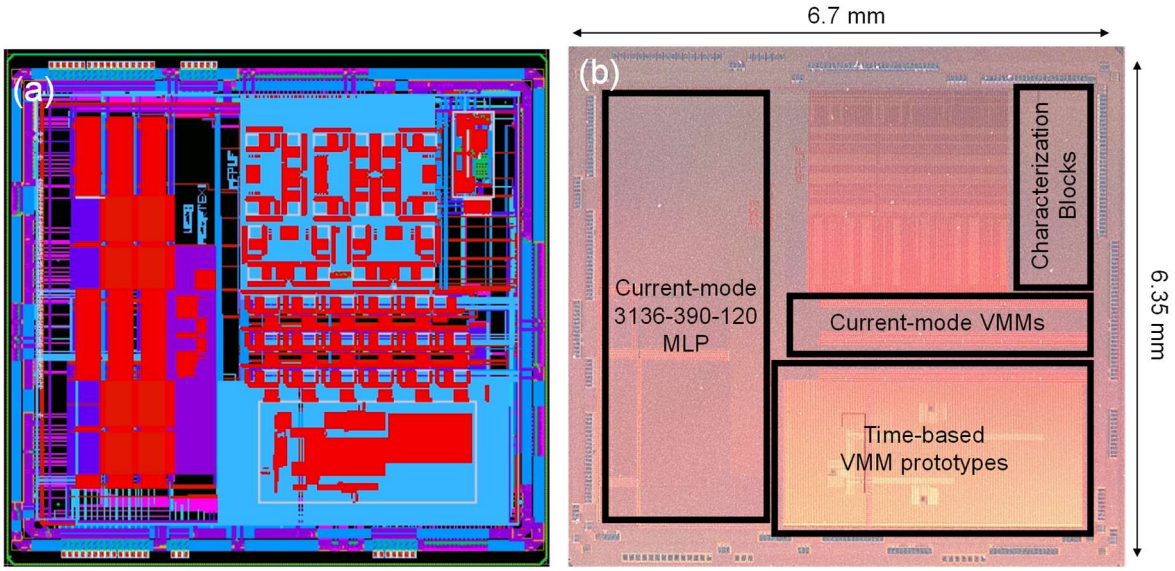


Fig. 40: High-speed image classifier and VMM prototypes in 55 nm CMOS: (a) layout and (b) micrograph of the fabricated chip.

Finally, a general-purpose neuromorphic architecture with industrial 1T1R analog memories is also designed. The architecture of this chip is based on our recent work, aCortex [67], which is designed primarily for eFlash technology. Several unique features of aCortex are exploited in this chip as well, including a configurable chain of buffers and data buses, simple and efficient instruction set architecture, and programmable quantization range. However, the most central feature of this design, which is the focus of this section, is configurable mixed-signal computing VMM blocks based on analog memories. Peripheral analog circuits are shared among a large array of VMMs that enable compact implementation and energy-efficient operation.

The overall architecture of this chip is shown in Fig. 41. The major on-chip components are a 32kb SRAM-based main memory, router, configurable chain of digital input buffers, two flexible 2D arrays of 36×44 VMM blocks, and output neuron blocks. In our design, input DACs, ADCs, and therefore activations all have a 4-bit resolution. The controller and memory for storing instructions are implemented offchip in this design. The chip could be used to accelerate a wide range of neural network inference models. The training of the models is performed ex-situ, i.e., weights are precomputed on a server. In the tuning phase, we transfer the weights into the conductance of memristive devices and store them in 1T1R arrays. Then, the chip is ready to perform the inference task: input data are loaded to the main memory, the controller executes necessary operations to perform the inference task, and the inference results will be computed and stored on the chip. The inference is performed layer by layer by sequentially reading the input from the main memory, loading them into digital buffers using the router, activating proper VMM and neuron blocks to perform the target dot-product operation, and then the results are transferred back to the main memory via the router. More detailed information on how the instructions, data management, and digital buffers are discussed in Refs. [67].

Future works will focus on the experimental characterization of the fabricated chip in a 65 nm CMOS process. Fig. 42a shows the micrograph of the entire chip, and Fig. 42b shows the adapter designed to interface the chip with the general-purpose characterization board.

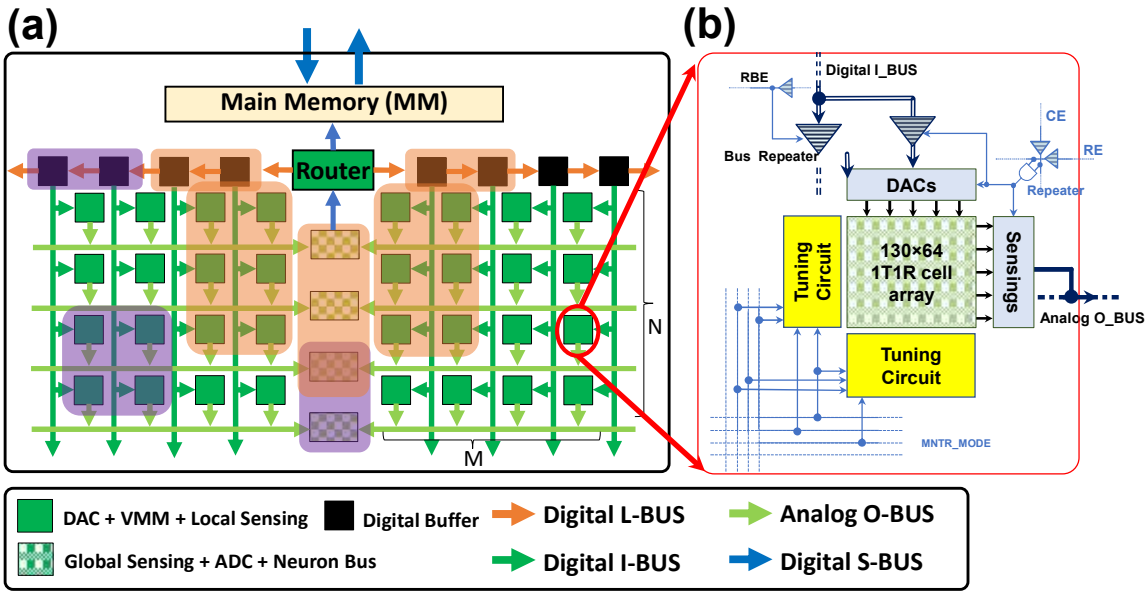


Fig. 41: The top-level architecture of Cerbero and its main computational core. In this design, M=36, and N=45. Digital L-BUS is the data path between the router and digital buffers (where data is loaded before the computation is performed). Digital I-BUS is the vertical data path that broadcast loaded data to DACs and VMMs. Analog O-bus are horizontally shared lines that connect the outputs of VMM blocks in each row. Digital S-BUS is the data path from the neuron blocks to the router and main memory. (b) VMM computing block including a 130×64 crossbar, tuning circuits, DAC arrays, sensing circuits, and logics that enable and disable the operation of the circuit in different modes. Paritucaly, CE is the column enable signal that enables a column of VMMs, and RE is the row enable signal that enables a certain row of VMMs. RBE is also an enable signal that allows us to cut the I-BUS at a certain row and avoid moving the data into unneeded VMMs.

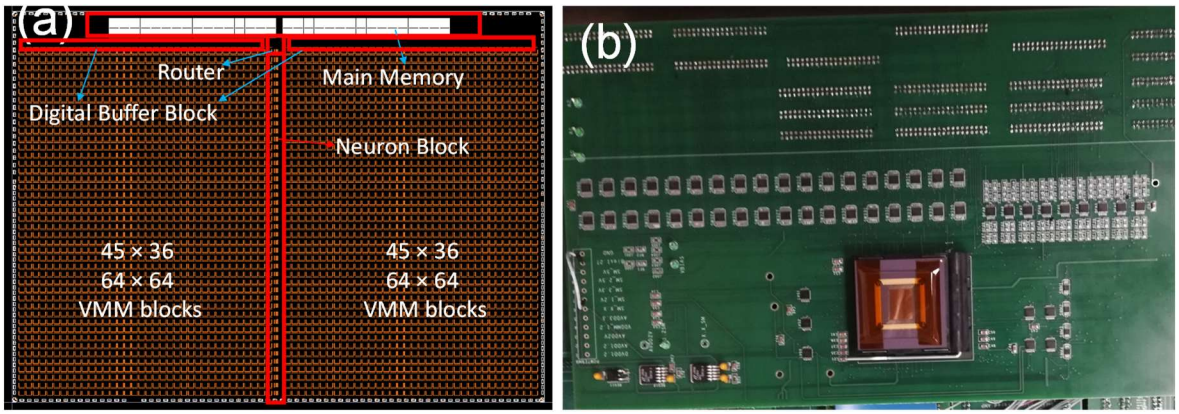


Fig. 42: (a) Micrograph of the fabricated chip. (b) Adapter to interface the chip with the measurement setup.

2.8. A Novel Temperature Compensated Current reference in 65 nm CMOS

The essential components of mixed-signal neuromorphic circuits are analog CMOS circuits, such as transimpedance amplifiers and current conveyors, which require on-chip distributed precision references. Precision current references directly impact the computation accuracy of neuron circuitry and, ultimately, the performance of the neurocomputing hardware. Besides the standard low power and small form factor features, neuromorphic circuits impose some exclusive requirements on the design of the current reference. Specifically, while a slight amount of mismatch due to process variations could be compensated or tolerated within the neuromorphic network, the variations in temperature and supply voltage are difficult, if not impossible, to counterbalance at the high-level. For example, the gain of a current conveyor-based neuron [130] is directly determined by its bias currents. A slight mismatch in the current can be mitigated by fine-tuning the synapses, but random dynamic shifts through temperature and supply voltage (during the runtime) could potentially generate large errors. Further, the complexity of these massive systems prevents the usage of trimming/calibration circuits or off-chip elements in the analog reference blocks.

The most common solution to generate a CWT reference is by summing up carefully designed complementary to absolute temperature (CTAT) and PTAT current references [191]. This could be realized, e.g., by using the voltage drop produced by the base-emitter voltage of parasitic bipolar transistors and the threshold voltage of MOSFETs [192] on a resistor. This technique achieves a temperature coefficient (TC) of 130 ppm/°C for a 10 μA reference in 0.005 mm^2. A similar idea is improved in [193] by reusing cascode transistors and combining it with a digital trimming circuit to compensate for the variations, which obtains 80 ppm/°C while consuming an enormous ~80 μw of power and 0.065 mm^2. Another solution to generate

a CWT current is to divide a carefully designed PTAT voltage (from the threshold of MOSFETs) over a PTAT resistor [194], which requires a resource-hungry voltage reference and a transconductance amplifier to eliminate the 100% line sensitivity. More complicated trimming techniques, e.g., based on nonvolatile memories [195], have also been proposed to enhance the temperature coefficient and variations at the expense of extra fabrication cost and power/area consumption. Several nano-ampere range references are also recently proposed [196-198] for energy-harvesting systems, which operate at very low voltages and deep weak inversion regimes. These designs often underperform at high temperatures and require inevitable calibration to reduce variations and the temperature coefficient.

Hence, the majority of previous works either are bulky, consume a large power consumption or require tremendous calibration to achieve a desirable TC. Unlike previous works, which are based on bipolar transistors or summation of PTAT and CTAT currents, a new methodology is developed in this section that relies on finding the optimum overdrive of a MOSFET through a modeling strategy. A compact, low-power microampere-range current reference is built based on this idea, which tackles the temperature and supply variations through a multi-threshold circuit design technique and is befitting for the requirements imposed by mixed-signal neuromorphic systems.

2.8.1. Proposed CWT Current Reference Circuit

High output impedance requirement is a primary feature of any current reference. Hence, the output current of a properly conditioned transistor (or cascoded transistors) could efficiently serve this purpose. In this elementary circuit, the quality of the generated reference current almost entirely depends on how the overdrive voltage of the transistor suppresses the variations in process, supply voltage, and temperature.

In sub-micron technologies, the square law approximation, i.e., $I_{\mathrm{d}} = (\beta/2)\, {V_{\mathrm{ov}}}^2$ (parameters have their usual meaning), fails to thoroughly predict the temperature dependency of the drain current, particularly at the onset of strong inversion in which V_{ov} is relatively small. This observation is confirmed in Fig. 43a that shows the simulated optimal overdrive voltage needed to generate a perfect CWT current reference in a 65 nm CMOS process. Using the conventional square-law approximation, we anticipate that, in order to produce a CWT current, the overdrive voltage should be monotonically increased with temperature as mobility reduces with temperature in silicon [199]. However, we observe that as the overdrive become smaller, the slope of the optimum curve $V_{\mathrm{ov,op}}(T)$ becomes smaller and ultimately negative, roughly 80 mV before the device enters the subthreshold regime. As a result of this large discrepancy, we develop a rather unconventional modeling methodology to analyze our proposed circuit and find the optimum design parameters.

A single curve of $V_{\mathrm{ov,op}}(T)$ that corresponds to a specific current density in Fig. 43a can be modeled via a multi-order polynomial function. For example, Fig. 43b shows the modeling results for a specific case of I_{ref} = 2.5 μA. As expected, the higher order of the polynomial used, the better the goodness-of-fit (and more resources need to generate it) and consequently, the lower the temperature coefficient. Remarkably, it is observed that we can achieve TC~80 ppm/ºC by using a second-order compensation in which $V_{\mathrm{ov,op}}(T)$ is modeled by $a_0 + a_1 T$. In other words, for a given reference current, by optimizing a_0 and a_1 with respect to TC, we can use this simple function to capture the essence of all complex temperature-dependent parameters of a device and still achieve <100 ppm/ºC, which is adequate in many other applications.

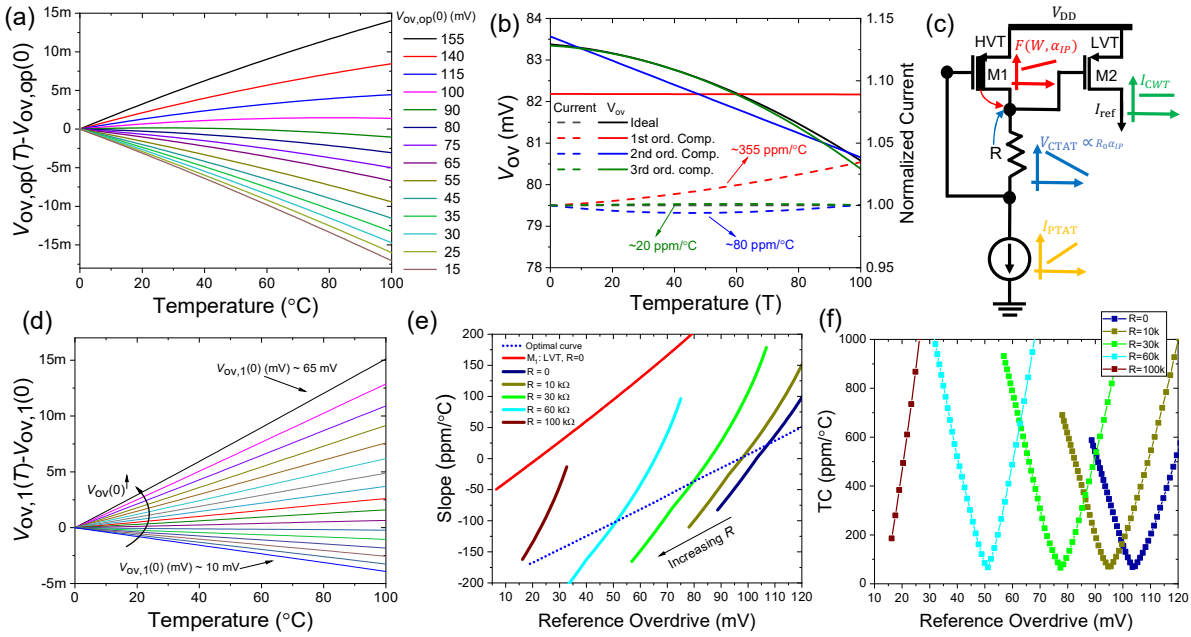


Fig. 43: (a) Simulated normalized optimal overdrive voltage ($V_{ov,op}(T)$- $V_{ov,op}(T=0)$) for achieving a CWT current using a PMOS. Here, a fixed 2.5 µA current is sunk from a long-channel PMOS (L=2 µm) in a diode-connected structure, and the width of the device is altered to change the $V_{ov,op}$ (0). Note that the channel-length modulation, drain-induced barrier lowering, etc., i.e., the contributions from the drain have an infinitesimal impact (~10 ppm/°C) on the results and can be easily ignored. (b) The modeling of optimal overdrive voltage obtained from (a) for the case of I_{ref}=2.5 µA, W(µm)/L(µm)=7.6/2 using a polynomial function. Without any compensation (i.e., driving a device with a constant gate voltage that corresponds to 2.5 µA at 0 °C), we get >4300 ppm/°C temperature variations. The first order compensation that crosses the optimal $V_{ov,op}$ curve in a single point reduces the TC to ~355 ppm/ °C (effectively, only the temperature dependency of the threshold voltage is compensated). The second-order compensation (using a_0+a_1T) crosses the optimal curve in two points and diminishes the TC to ~80 ppm/°C. A third-order compensation scheme using a_0+a_1T+a_2T^2 reduces the temperature coefficient further to ~20 ppm/°C. (c) The simplified schematic of CWT reference generator. (d) The change in the overdrive of an HVT device that sources a PTAT current versus temperature for various reference overdrives (V_{ov} (T=0)). (e) The slope of the fitted overdrive voltage of M_1 versus the overdrive reference $V_{ov,2}$ (T=0) in various design cases. The blue line (optimal curve) corresponds to the fits to $V_{ov,op}(T)$. The red corresponds to the case when both transistors are LVT and R=0. Note that by increasing R, the gap between the red curve and the optimal curve widens even more. The curves obviously never cross each other, even in the subthreshold regime. HVT transistor is used in other curves (with various values of R). (f) The temperature coefficient of the output current reference as a function of the reference overdrive corresponding to the cases studied in panel (e).

Fig. 43c shows the lightest approach of generating $V_{ov,op}(T)$ (for M_2) and consequently a CWT current. M_1 and M_2 are both operated in the strong inversion regime, and the overdrive

of M_2 is given by $V_{ov,2} = V_{ov,1} - RI_{PTAT} + \Delta V_{th}$, where ΔV_{th} is the difference between the threshold voltage of the devices. Simulation results also show that the overdrive of M_1, which sources a PTAT current, is (almost perfectly) linearly proportional to temperature (Fig. 43d), i.e., $V_{ov,1}(T) \approx b_0 + b_1 T$ with b_0 and b_1 are controlled with the sizing of the device (M_1).

Prior to explaining why this circuit generates a CWT current, let us clarify few assumptions. We use $T_0 = 0$ as a reference point and $R(T) \approx R_0(1 + \alpha_R T)$, $\Delta V_{th}(T) = \Delta V_{th,0}(1 + \alpha_{vt} T)$, and $I_{PTAT}(T) = I_{P,0}(1 + \alpha_{IP} T)$, where $\Delta V_{th,0}$, α_{vt}, and α_R are process parameters, R_0, W_1, $I_{P,0}$, α_{IP} are design parameters. $\Delta V_{th,0} = 77$ mV, $\alpha_{vt} = -59$ ppm/°C, $\alpha_R = -131$ ppm/°C (P-type non-salicided poly resistor) and simulations are based on a 65 nm process, $L_{1,2}$=2 µm is used to minimize the variations, flicker noise, and the output conductance. For simplicity and reducing the number of design parameters, a PTAT current reference is initially designed (details are provided in the next section) with $I_{P,0}$=0.8 µA and α_{IP}=2.8×10^{-3} °C^{-1}.

The main purpose of the proposed circuit is to employ the PTAT current source, resistor, and M_1 for generating a pseudo-optimal overdrive curve (with respect to temperature) in M_2. In this circuit, if R=0 and both M_1 and M_2 were both LVT devices (i.e., $\Delta V_{th} = 0$), $V_{ov,1}(T) = V_{ov,2}(T)$ assuming no mismatch. In such a case, it is evident that when M_1 sources a PTAT current, M_2 can not generate a CWT current under any circumstances. But it will clarify our idea if we study how $V_{ov,1}(T)$ changes when we change $V_{ov,1}(0)$ (by sizing of M_1) with respect to the optimal curve $V_{ov,opt}(T)$ which was previously modeled. The red line in Fig. 43e shows the fitted slope $V_{ov,1}(T)$ (i.e., b_1) versus b_0. Comparing it with the modeled optimal curve and considering that $V_{ov,1}(T) = V_{ov,2}(T)$, we find that the curves do not cross at any point when the initial reference curves are the same, essentially meaning that if M_1 is LVT MOSFET, there is no optimal sizing of M_1 for generating an optimal overdrive in M_2. Note that the gap

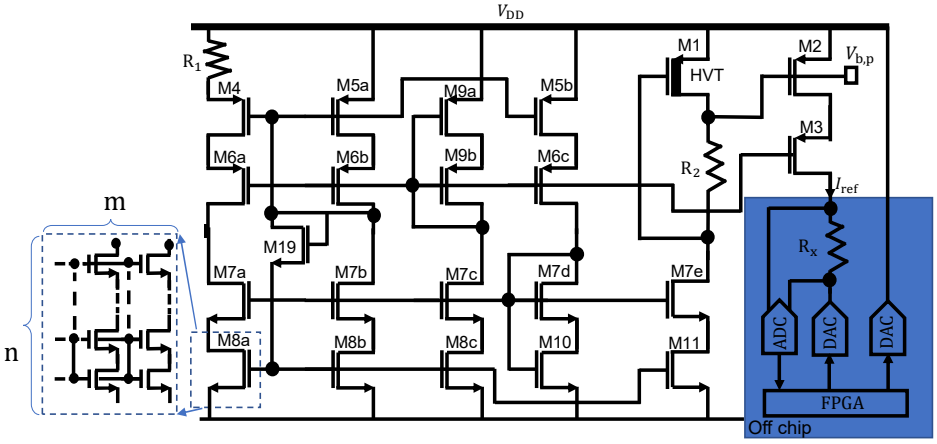
between the two curves can only be reduced by diminishing the temperature coefficient of the PTAT current (and they will collapse if M_1 sources a CWT current). Now, if we use an HVT device to implement M_1, the red curve shifts by ΔV_{th} to the right and creates a crossover point. Note that $V_{\mathrm{ov},2} = V_{\mathrm{ov},1} + \Delta V_{\mathrm{th}}$ and since α_{vt} is very small, the slope changes negligibly. Moreover, by increasing R, we shift the curve to the left and change its slope as well. Effectively, we obtain $V_{\mathrm{ov},2} \approx b_0 + b_1 T - R_0(1+\alpha_{\mathrm{R}}T) I_{\mathrm{P},0}(1+\alpha_{\mathrm{IP}}T) + \Delta V_{\mathrm{th},0}(1+\alpha_{\mathrm{vt}}T)$, which is the desired second-order linear compensation, since $\alpha_R \alpha_{IP} T^2$ is negligible in the practical temperature range. Note that, temperature-wise, this circuit has many optimum pairs of (R, W_1). The simulation results in Fig. 43f show the TC of the output current for various resistance values and confirms that the optimal point (the minimum TC) occurs almost perfectly at the crossover of points of Fig. 43e, verifying the idea and the fact that linear modeling of the optimal curves works as expected. R_1 sets the reference overdrive of both M_1 and M_2, and its value is selected to optimize the performance with respect to process variations.

One important note is that this simple circuit also supports rudimentary features that ensure suppression of supply voltage variations as well. Assuming that the PTAT reference has a negligible supply voltage dependence, the overdrive of the output device is independent of the supply voltage. In addition, using only (same-length) PMOS devices eliminates global variations, cancels out any systematic mismatch, and leads to overall better resiliency toward local variations.

2.8.2. Fabrication and Measurement Results

Fig. 44 shows the complete schematic of the fabricated circuit in a 65 nm CMOS process. The design primary includes a beta-multiplier circuit (R_1, $M_{4a,b}$, $M_{6a,b}$, $M_{7a,b}$, $M_{8a,b}$) that

generates a supply-insensitive PTAT current, essentially controlled by the ratio of M_5 and M_4 and the resistance of R_1 [191]. M_{19} is used to prevent any start-up issues, and $M_{9a,b}$, M_{7c}, M_{8c}, M_{5b}, M_{6c}, M_{7d}, M_{10} are used to generate the bias voltage for the cascode transistors in the PTAT generation branches. M_{11} and M_{7e} copy the PTAT current into the CWT generation branch. M_1, M_2, and R_2 generate the CWT current. The devices are implemented in an array fashion to exploit standard layout techniques that minimize the impact of variations. The length of all cascode devices is 1 µm, and the rest of the devices have a channel length of 2 µm. Note that the sizing of the devices provided in the Table is performed targeting both minimum occupied area and process variations. The process includes super low threshold, LVT, and HVT devices. All devices in this circuit are implemented with LVT MOSFETs ($V_{\mathrm{th0,n}}$~0.43 V and $V_{\mathrm{th0,p}}$~0.35 V) except for M_1 that is an HVT MOSFET ($V_{\mathrm{th0,p}}$~0.43 V).



	W (µm)	m	n
M1	0.36	15	1
M2	0.38	20	1
M3	0.23	36	1
M4	0.2	28	3
M5	0.38	8	3
M6	0.23	36	1
M7	2	2	1
M8	0.24	30	3
M9	0.23	36	2
M10	0.23	10	3
M11	0.24	10	3
R1	30 kΩ		
R2	27 kΩ		

Fig. 44: (a) The schematic of the fabricated current reference. From the left side, the first two branches generate a PTAT current, the next two branches produce the cascode biases for the circuit. The last two branches implement the CWT generation. The table shows the size of the devices used in the circuit. Blocks highlighted in blue are used to measure the output current and implemented offchip.

The supply voltage (1.2 V, nominal) is provided to the chip by a 16-bit digital-to-analog converter (AD5541A) buffered by an LT1637 operational amplifier. The measurements are performed using a 16-bit 5 Msps analog-to-digital converter (LTC2325-16) that measures the

reference current in the form of a voltage drop across a thin film ~43 kΩ resistor, which is biased at 0.5 V via another buffered DAC.

A total of 4 dies from one wafer are packaged for measurements. TC measurements are conducted by baking the samples from 0 to 100 °C using a Eurotherm temperature controller. The measured TCs are 253, 184, 386, and 283 (with an average of 276) ppm/°C for each chip. More statistically meaningful results are obtained from 500 runs of MonteCarlo simulations in various corners. The average (standard deviation) of TC in TT, SS, FF, SF, and FS corners are 78 (12.5), 85 (21), 113 (24), 90 (15), 77 (18), respectively. We speculate the discrepancy between experiment and simulation results stems from the inaccurate Spectre models of resistors with minimum width (which we were not aware of them during the Tapeout), which slightly shifted the optimum point.

Fig. 45a shows the measured change in the reference current with respect to variations in the supply voltage at room temperature and indicates the proper functioning of the circuit for 0.95 V< V_{DD} <1.3 V and the average line regulation of 1.9 %/V. This is in agreement with simulations that show that a 1% change in reference current occurs at 0.93 V and 1.41 V, in average. Simulated variations in the average current in TT, SS, FF, FS, and SF corners are 2.1%, 2.4%, 1.9%, 2.1%, and 2.4%, respectively. Fig. 45b shows the packaged chip, and Fig. 44c depicts the micrograph of the zoomed-in view of the portion fabricated chip. The power supply rejection ratio is 149 dB at 1kHz. Table 4 compares the results of the measurements with state-of-the-art current references.

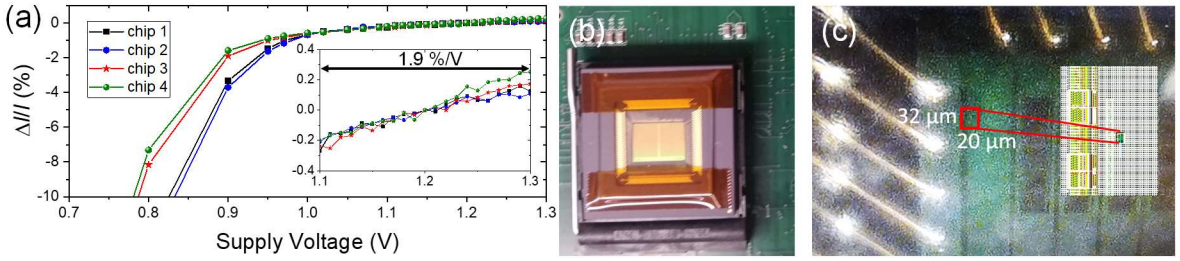


Fig. 45: (a) Measured change in the output current, i.e., $(\Delta I/I_{\text{ref}})\times 100$ with reference computed at the room temperature and 1.2 V, versus the supply voltage. The inset shows the zoomed-in view to the portion of the figure. (b) The packaged mixed-signal neuromorphic chip including the current reference. (c) Zoomed-in view to the portion of the chip that includes the current reference block. Inset shows the magnified layout of the reference block.

Table 4: Comparison with previous works.

	CICC [192]	*TCASII* [193]	*JSSC* [194]	*JSSC* [195]	*ESSCIRC* [196]	*SSCL* [197]	This work	
Technology (µm)	0.18	0.35	0.18	0.5	0.18	0.065	0.065	
Sup. Voltage (V)	2.4-3	1.9-3.6	1-1.2	>2.3	1.2	>0.4	1.2	
Power (µW)	>48	~88	>32.7	NA	23×10^{-6}	3.4×10^{-6}	15.8	
Ref. Current (µA)	10	16	7.81	16÷50	20×10^{-6}	1.2×10^{-6}	2.5	
Temp. Range	-40 : 80	0 : 110	0 : 100	0 : 80	0-80	-20 : 60	0 : 100	
TC (ppm/ºC)	130	80	24.9	<130	780	470	88*	276
Trimming	NA	Digital	NA	NVM	Digital	Digital	NA	
Line Reg. (%/V)	0.5	4	100	<1	0.58	2.5	1.9	
Area (μm^2)	5k	65k	230k	>60k	48.4k	8k	640	

Most previous works are relatively complex and have a large number of design variables [193]. In our work, we model the complex mathematical temperature dependency of nanoscale FETs at the onset of strong inversion and show that second-order semi-linear modeling yields <100 ppm/°C. A very compact topology to implement our idea is then proposed. A small number of transistors is also helpful in terms of noise performance [197]. More sophisticated circuit techniques, e.g., replacing the resistor with a switch capacitor equivalence, could improve the performance at the cost of an extra area, which we try to avoid here. The idea of biasing a MOSFET in the vicinity of its zero TC point [200] is also interesting, except that it requires a carefully designed extensive bandgap reference and calibration, and the zero TC point occurs at very large overdrives (~300 mV at a 65 nm process).

In this chapter, we demonstrate a compact and low-power compensation technique for the design of precision current references. Instead of delving into complicated analytical models of temperature dependency of the drain current of MOSFETs, we use a novel modeling framework to generate a pseudo-optimum overdrive curve for a MOSFET. A multi-threshold circuit, including only two transistors and a resistor, is then used to produce that voltage, essentially acting as a PTAT to CWT converter. A graphical analysis methodology is provided for finding the optimum design parameters. The circuit is fabricated in a 65 nm CMOS process, and experimental results from 4 samples show the average TC of 276 ppm/°C and ~1.9 %/V line regulation.

2.9. A General-Purpose Experimental Setup for Characterization of Mixed-Signal Circuits and Systems

We have designed, fabricated, and successfully tested a general-purpose standalone measurement setup that can be used to characterize the mixed-signal neuromorphic chips, which were discussed in previous sections. Fig. 46 shows the top-level architecture of this setup. The FPGA is already programmed with the necessary functions to access and control the components onboard and, more importantly, to interface with the host board. The operation of the board begins with the user running his/her codes in the MATLAB environment installed on the personal computer. The computer sends the commands over the Ethernet port to the FPGA. Subsequently, the FPGA receives the command and takes the necessary steps to perform that command. After the command is executed, either data or an acknowledge flag of a successful execution is transmitted to the user.

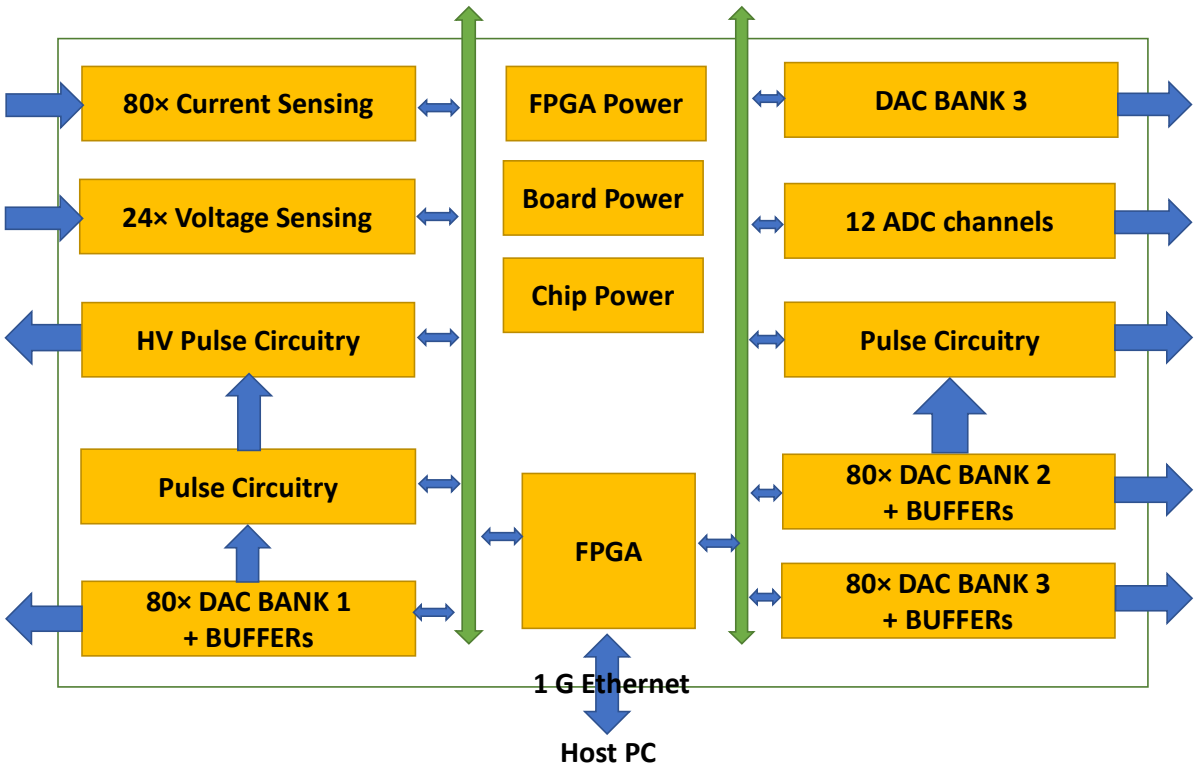


Fig. 46: The overall structure of the general-purpose measurement setup.

Among the exciting features of this system, the following are noteworthy: 1) a high-speed 1Gbps interface with a Host PC controlled by a MATLAB user-interface program; 2) 50 high-voltage pulse generator with variable pulse amplitudes as high as 11V, and variable pulse widths as fast as 10 ns, 3) 50 low-voltage pulse generators with variable pulse amplitudes as high as 3.5 V, and varying pulse widths as fast as 10 ns, 4) 3 banks of 80× 16-bit 1 MSPS DACs (which are serially programmable within each bank), 5) Buffering amplifiers are used which provide output currents as high as 10 mA, 6) 12× directly accessible 16-bit 1 MSPS ADC channels, 7) 80× current sensing circuits with adjustable range, 8) 100 digital IO pins from FPGA for extra digital control and data transfer, 9) Provides multi-power rails (1.2V, 1.5V, 2.5V, 3.3v, 5V, 12V) for powering device-under-test. The current flowing in two of the power rails is measurable. Fig. 47 shows the assembled system while interfacing with an adapter designed to test the ConvNet accelerator.

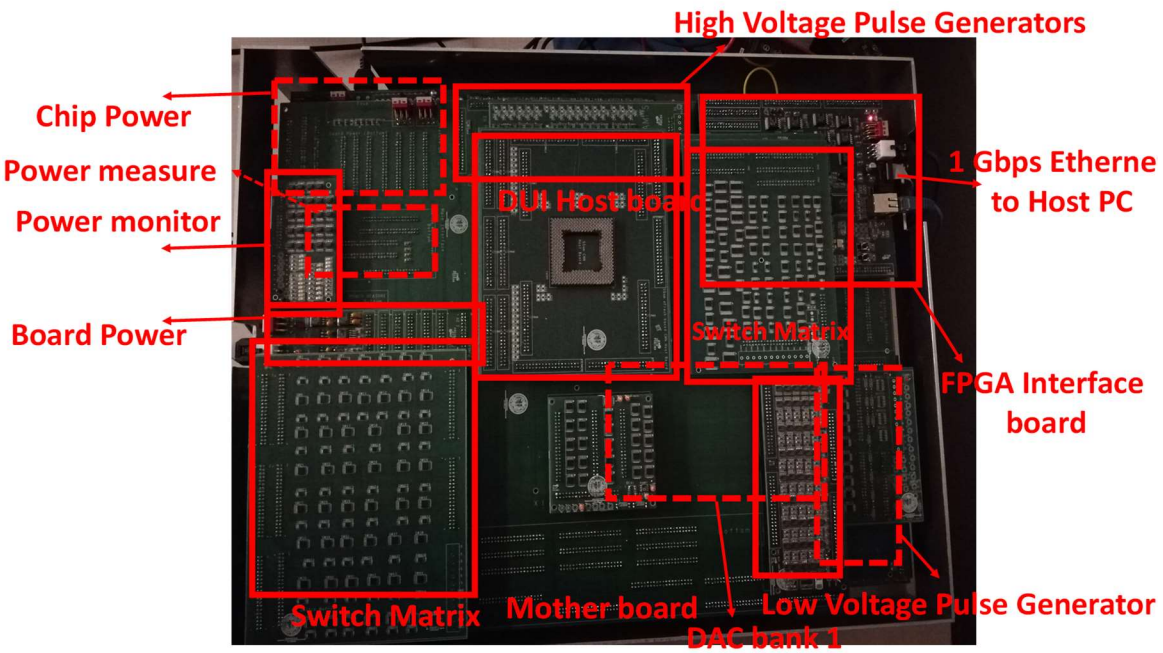


Fig. 47: The overall structure of the general-purpose measurement setup that includes 3 layers of multi-layer PCBs. All circuits in this board, including the FPGA board, are custom designed.

2.7. Summary and Future Works

Analog-grade nonvolatile memories such as eFlash memories and metal-oxide memristors are enabling components for mixed-signal neuromorphic circuits. Such circuits implement the dot-product operation, the most common operation in any artificial neural network, in the analog domain or physical-level, enabling the design of energy and area efficient neuromorphic hardware. In this chapter, we discussed various device candidates and topologies for implementing dot-product operation in the mixed-signal domain. The main focus was on resistive memories and 2D eFlash memories, and current-mode VMMs. We have also conducted a preliminary study of using the extremely compact and scalable 3D NAND technology [142,143] without a need for modifying the arrays due to its inherent compatibility with time-domain VMMs [164]. A rigorous analysis of these circuits, which

takes various nonidealities (drain-induced barrier lowering, capacitive coupling, charge injection, etc.), promises a high area and energy efficiency. Such mixed-signal VMM circuits are particularly appealing for accelerating massive deep neural networks whose weights cannot be stored easily on a single chip with 2D memories. Future work should focus on the experimental demonstration of such circuits and analog characterization of 3D NAND memories. Further, we have also preliminary studied the possibility of improving the dynamic range and increasing the mismatch resiliency in active memristive crossbars [144]. Accordingly, a 2T1R memristive array is designed that mitigates these issues and promises high energy efficiency when used in the time-based VMM topology and aggressively-scaled resistive memory technology. A very useful future work is to study the prospects of each stack and topology for various computing applications, e.g., building fully-analog neural hardware, general-purpose accelerators, scientific computing machines, etc., in a single all-in framework that considers energy efficiency, accuracy, area, and throughput.

The second and third sections of this chapter were focused on analog-grade memristive technology. Specifically, we demonstrated the largest passively-integrated memristive crossbar to date (supported with proper statistics) and showed a 64×64 passive crossbar circuit with ~ 99% functional nonvolatile metal-oxide memristors featuring etch-down patterning and a low-temperature budget based on a foundry-compatible fabrication process. Perhaps, the most important achievement in this project is the reported <26% normalized variation in memristor switching voltages, which was sufficient for analog programming of the array with <4% relative tuning error. We believe that the near-term work should focus on improving technology to increase yield and reducing device variations, decreasing write and operating currents of memristors, and ultimately demonstrating practical fully-integrated hybrid circuits,

e.g., with back-end-of-line fabricated memristors on top of the CMOS subsystem that would outperform purely-CMOS counterparts. Theoretical efforts should focus on the development of holistic circuit and algorithm techniques for coping with device variations and faulty devices. It is expected that the goal of decreasing the cell currents could be achieved by reducing leakages within the device and between neighboring lines, e.g., by patterning the active switching layer and scaling down device feature sizes.

In the third section of this paper, we investigate the role of uniformity in passive crossbar circuits using all-embracing simulations of general VMM blocks and representative neural networks. In this study, we reported three post-fabrication methods that increase the resiliency of memristive neuromorphic circuits toward the half-select disturbance. These techniques were able to recover the accuracy drop in two major neuromorphic networks when built with our 64×64 memristor technology. We believe that experimental validation of these techniques is a critical future study to strengthen the claims made on the prospects of 0T1R technology in neuromorphic circuits. Finally, at the network level, more creative solutions may exist, which are worth investigating and further research. In this section, a comparative study on the scaling prospects of 0T1R and 1T1R technology was conducted assuming a fixed CMOS technology node (65 nm). A more generalized study that also takes into account the scaling of CMOS circuits is an exciting and valuable future work.

The investigation of two important nonidealities in memristive current-mode VMMs [162], i.e., IR-drop and device nonlinearity, in section 2.4 indicates a significant challenge of this technology. Our findings showed that when the crossbar size is scaled up, the optimum interconnect wire conductance should increase quadratically to preserve the computing precision, and the differential scheme is imperative for temperature insensitive operation and

reducing the IR-drop effect. Further, these issues were partially mitigated by a balancing technique that minimizes the error distribution by optimizing the tuning voltage and bootstrapping approach that tackles the IR drop issue by increasing the effective wire conductance at the cost of monopolizing other CMOS metal layers. Both of these techniques are applied in the circuit design and operating levels.

In the following, we designed two critical circuits to enable energy-efficient computing with low-power eFlash memories. Currently, these devices are arguably the most energy-efficient option for implementing analog synapses due to their excellent retention characteristics and realizability at deep subthreshold <100 nA regimes. First, a compact sensing circuit is designed in a 55 nm CMOS process, leading to a dramatic reduction power/area overhead of sensing circuits in analog eFlash-based VMMs [163]. Second, an internally-analog externally-digital VMM is developed using the proposed sensing circuit and a current-mode algorithmic ADC, which exploits analog-grade eFlash memories for unit current generations and offset compensations, breaking POp/J energy efficiency regimes [164]. The sensing circuit is utilized as a local sensing circuit in the fully-analog multilayer image classifier chip. The proposed internally-analog externally-analog VMM is also used in the design of our recently proposed aCortex architecture [38], which is a general-purpose neuromorphic accelerator.

Finally, we presented the design and fabrication of three mixed-signal energy-efficient neuromorphic networks. First, the architecture of a network-specific DNN accelerator was discussed. The fabricated network is a 6-layer convolutional neural network, which includes ~1.5×10^6 training parameters and ~3×10^3 neurons. We presented unique features of this design, including TDM architecture, NOR-flash-based DAC, local and global folded wiring,

novel design of tuning circuits, programmable activation function slope, individual access to each neuron output for advanced training schemes, and more importantly, massive read/write/erase parallelism, which enable sub-minute high precision tuning of the entire chip. Then, the design of an extremely high-speed 3-layer multi-layer perceptron based on 55 nm eFlash memories was presented. Simulation results indicate that the chip can classify MNIST images with ~98% accuracy, and the classification of one pattern takes ~100 ns and consumes 0.25 nJ—4.3%, 10×, 100× better than our previous implementation [70] based on 180 nm eFlash technology. The improved performance originates not only from exploiting a more advanced process node but also from employing more optimum peripheral circuits. This tapeout also included prototypes of energy-efficient current-mode and time-based VMMs. Finally, a general-purpose DNN accelerator in 65 nm CMOS based on 1T1R memories was presented. The major on-chip components are a 32kb SRAM-based main memory, router, configurable chain of digital input buffers, two flexible 2D arrays of 36×44 64×64 2-quadrant VMM blocks (a hence a total of ~26×10^6 1T1R cells), and output neuron blocks. The chip could be used to accelerate a wide range of neural network inference models. We discussed various aspects of this system, including VMM topology, tuning circuits, and analog peripheries.

Future work should focus on electrical measurement and characterization of the fabricated chips. We have already designed a generic experimental setup for this purpose. The backbone and main features of this system were illustrated in section 9. For each board, a PCB adapter is designed to interface with the setup and to some chip-specific functionalities. Proper functionality of the experimental setup and the interface boards have also been validated.

3. Neuromorphic Computing

The development of efficient specialized hardware for stochastic neural networks, especially for Boltzmann machines, which have become the state-of-the-art approach for solving many problems in machine learning, information theory, and statistics, is of great importance. By now, there have been many demonstrations of efficient hardware for dot-product computation, the most common operation in stochastic (and virtually any other) neural networks. Separately, there were numerous hardware implementations of stochastic neurons, which are a unique feature of Boltzmann machines. However, an efficient and scalable hardware solution combining both functionalities is still missing. Here we report compact, fast, energy-efficient, and scalable stochastic dot-product circuits that may use either of two types of memory devices – passive integrated metal-oxide memristors and embedded floating-gate memories. The circuit's high performance is due to mixed-signal implementation, while the efficient stochastic operation is achieved by utilizing the circuit's noise, which could be intrinsic or extrinsic to the array of memory cells. The functionality of the proposed approach is experimentally verified by implementing a Boltzmann machine with 10 input and 8 hidden neurons.

The second focus is the acceleration of the neurooptimization tasks with mixed-signal circuits. The increasing utility of specialized circuits and growing applications of optimization call for the development of an efficient hardware accelerator for solving optimization problems. Hopfield neural network is a promising approach for solving combinatorial optimization problems due to the recent demonstrations of efficient mixed-signal implementation based on emerging nonvolatile memory devices. Such mixed-signal accelerators also enable very efficient implementation of various annealing techniques, which

are essential for finding optimal solutions. In this chapter, we study various annealing techniques and their efficient implementation to improve the performance of Hopfield neural networks. We proposed an efficient implementation of simulated annealing, chaotic annealing, and novel weight annealing with analog-grade memristive crossbars and eFlash memories. The hardware operation is successfully tested by experimentally solving weighted graph partitioning, maximum clique, vertex cover, and independent set problems and observing good agreement with simulation results.

2.10. Introduction

Computations performed by the brain are inherently stochastic [145-151]. At the molecular level, this is due to stochastic gating of ion channels of the neurons [147] and the probabilistic nature of transmitter release at the synaptic clefts [148]. Noisy, unreliable molecular mechanisms are the reason for getting substantially different neural responses to the repeatable presentations of identical stimuli, which, in turn, allows for complex stochastic behavior, such as Poisson spiking dynamics [146,149,150]. Though noise is always detrimental for conventional digital circuits, a very low signal-to-noise ratio (SNR) of neuronal signals [151] has been suggested to play an important role in the brain functionality, e.g., in its ability to adapt to changing environment [145,146], as well as for achieving low energy operation [152]. It is therefore not surprising that many developed artificial neural network algorithms rely on stochastic operations. For instance, probabilistic synapses could be used as the main source of randomness for reinforcement learning or as regularizers during training, significantly improving classification performance in spiking neural networks. In spiking neural networks, probabilistic synapses also allow relaxing the requirements for synaptic weight precision due to temporal averaging over a spike train.

Perhaps, the most prominent example is a Boltzmann machine, a recurrent stochastic neural network with bidirectional connections [153, 154], which belongs to the so-called energy-based models and can be viewed as a generalization of the Hopfield network [155,156]. In its simplest form, the Boltzmann machine is a network of N stochastic binary neurons. At each discrete time instance, the network is in a certain state, which is characterized by binary V_i outputs of its neurons. The network dynamics is arranged to model thermal equilibrium, at certain temperature T, of a physical system with energy E

$$E = -\sum_{i=1}^{N} V_i I_i\,, \qquad I_i = \sum_{j=1}^{N} G_{ij} V_j + I_i^{\mathrm{b}}, \tag{2.1}$$

where I_i and I_i^{b} are analog input and bias, which are typically represented by currents in the circuit implementation, while G_{ij} is a synaptic weight (conductance) between i^{th} and j^{th} neurons. The network state is updated by changing the state of the randomly chosen neurons. The probability of a neuron being switched to the digital state '1' with amplitude $V_{'1'}$ - in other words, turned 'on' or being activated - is a sigmoid function of its input, i.e.,

$$\Pr(V = V_{'1'}) = \frac{1}{1 + \exp\left(-\frac{I'}{T}\right)} \tag{2.2}$$

Here T is a dimensionless temperature, and I' is a normalized input current $I' = I/I_{\max}$, where $I_{\max}$ is the largest possible neuron input current, common for the whole network. (Note that though the neurons in the Boltzmann machine are partitioned into visible and hidden ones, for simplicity, we use the same notations for both types.) In the zero-temperature limit, the sigmoid in Eq. 2.2 becomes a step function, and the Boltzmann machine is similar to the Hopfield network in that given the proper set of weights (e.g., using symmetric, zero-diagonal matrix G), the total network energy is always decreasing when neuron states are sequentially updated at discrete time steps [155]. At non-zero temperatures, the same process would

typically lead to thermal equilibrium. For such a state, the snapshot of neuron outputs represents a sample from the thermal equilibrium probability distribution. In both zero and non-vanishing temperature regimes of Boltzmann machines, the process of simulated annealing, in which initially high temperature is gradually decreased over time, helps to escape local energy minima [156].

As a stochastic version of Hopfield networks, the Boltzmann machine, combined with simulated annealing, is a powerful approach for solving combinatorial optimization problems [156]. Moreover, such networks can be utilized to compute conditional and marginal probabilities by fixing the states of some neurons and sampling outputs of the unclamped ones. Such functionality is central for many Boltzmann machine derivatives, such as deep-belief networks and Bayesian model computing. The invention of the restricted Boltzmann machine (RBM) [153, 157] and efficient training algorithms [158] have led to its widespread use in many machine learning tasks, including dimensionality reduction, classification, and notably, collaborative filtering, for example enabling the best performance in the Netflix movie prediction challenge [159]. Furthermore, owing to similarities to Markov random fields, Boltzmann machines have found many applications in statistics and information theory [157].

The stochastic dot-product computation described by Eqs. 1-2 is the most common operation performed during inference and training in Boltzmann machines and their variants, and hence its efficient hardware realization is of utmost importance. In chapter 2, we discussed many demonstrations of high performance dot-product circuits, most importantly including analog and mixed-signal implementations based on metal-oxide memristors, and phase-change, and floating-gate memories, developed in the context of neuromorphic inference

applications. Analog dot-product circuits based on metal-oxide memristors have also been demonstrated in the context of neural optimization [60,161].

For Boltzmann machines, the stochastic functionality can be realized in neural cells, peripheral to the array of memory cells, rather than at much more numerous synapse locations, which somewhat relaxes the design requirements. Still, prior studies showed that even with a relatively large synapse to neuron ratio (~1,000) and deterministic dot-product functionality, the neuron circuitry may constitute a substantial part of the neuromorphic inference systems [70]. Because of such concerns, purely CMOS implementations, see, e.g., CMOS probabilistic gates [51] and CMOS-based Ising chip for combinatorial optimization problems [46], may not be very practicable.

The implementation overhead of stochastic functionality might be less of a problem for some memory devices, in which switching between memory states is inherently stochastic, e.g., ferromagnetic, phase-change, ionic and thermally-driven metal-oxide, and solid-state electrolyte devices. Unfortunately, many of such devices come with other severe challenges. For instance, an efficient implementation of large-scale dot-product computation is a major challenge for magnetic devices. The hybrid option of combining magnetic stochastic neurons with the already mentioned mixed-signal dot-products is not appealing because the interface typically compromises an extreme energy efficiency of spin-based computing with charge-based devices. The technology of magnetic devices is also relatively immature, judging by very few (and rather low-complexity) experimental demonstrations [165,166]. The biggest challenge for the remaining devices would be low switching endurance and cycle-to-cycle and device-to-device variations in switching characteristics.

2.11. Neurocomputing and Neurooptimization with Analog Memories

This section demonstrates a method of utilizing intrinsic and extrinsic current fluctuations in mixed-signal circuits based on analog-grade nonvolatile memories to implement scalable, versatile, and efficient stochastic dot computation. The deterministic version of such dot-product circuits has been extensively investigated due to their potentials for high speed, high density, and extreme energy efficiency – see, section I of chapter 2. Unlike many prior proposals [52, 53, 165,166], our approach is suitable for large-scale dot-product circuits and has no endurance restrictions for inference operation, which is typical for virtually all other proposals involving memory state switching [12,56,57,58,167,168]. We experimentally verify stochastic dot-product circuits based on metal-oxide memristors and embedded floating-gate memories by implementing and testing representative Boltzmann machine networks with non-binary weights and hardware-injected noise. We further demonstrate how scaling of synaptic weights during operation can be used for a very efficient implementation of simulated and chaotic annealing to improve functional performance.

3.2.1. Stochastic Dot-Product Circuit

Figure 48a shows the investigated current-mode analog circuit based on nonvolatile memories, in which vector-by-matrix operation is efficiently implemented on the physical level due to Ohm's and Kirchhoff's laws. For memristor-based circuits (Fig. 37b), the weights are encoded with device conductances so that the current flowing into the virtually grounded neuron is given by $\Sigma_i G_i V_{i,}$ and equations describe the network operation in Figure 2.48a. For the considered discrete-state networks, a crosspoint floating-gate transistor can be conveniently viewed as a switch connecting a current source to a neuron's input (Fig. 48c). The cell currents I_{cell} at voltage bias $V_{,1,}$ used at network operation are pre-set according to the

desired synaptic weight. The neuron input current is given by $\Sigma_i I_{\text{cell},i}(V_i)$, where $I_{\text{cell}}(V_{\text{'0'}}) \approx 0$ when digital '0' is applied to the cell's switch.

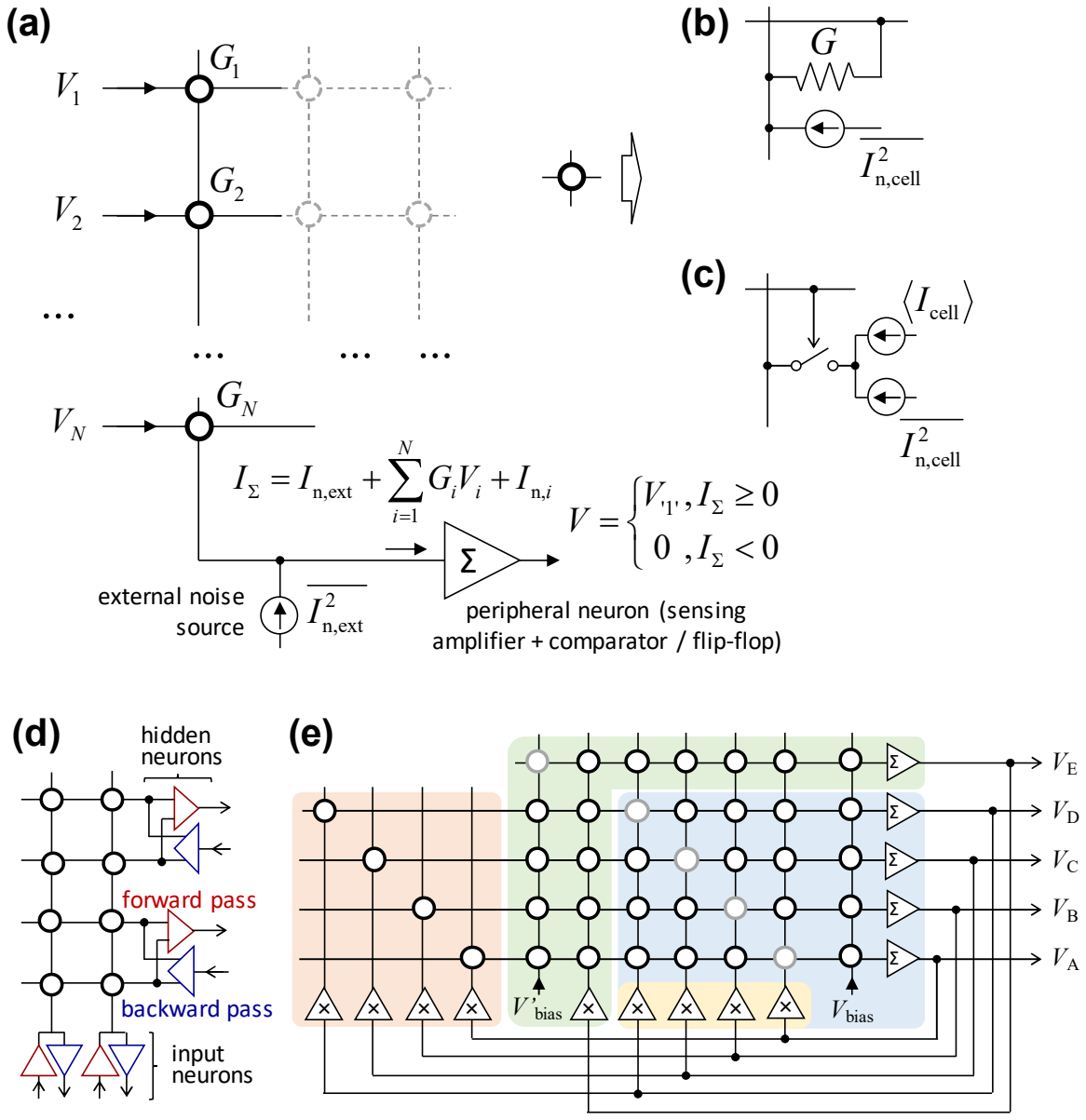


Fig. 48: Stochastic mixed-signal dot-product circuit and its application for neurocomputing and neurooptimization. (a) Circuit schematics for the design with current-mode sensing with crosspoint device implementation based on (b) memristors and (c) floating-gate memories. The equations in the figure correspond to memristor implementation, while their modified version for floating gate design is described in the text. (d) An example of the considered differential-pair Boltzmann machine implementation. (e) The implementation of generalized Hopfield neural network. The blue background highlights the baseline implementation. The yellow, green, and red backgrounds highlight additional circuitry for the proposed "stochastic", "adjustable", and "chaotic" approaches, respectively. The grey shaded circles

show synaptic weights, which are typically set to zero. Labels 'Σ'/ 'x' inside amplifier symbols denote summation/scaling. For clarity, panel a does not show bias currents, panels a and e show a single-ended network, while panel d shows a small two-input, two-hidden neurons fragment of the considered network.

The circuit noise is detrimental to the deterministic dot-product operation and, e.g., defines the lower bound on the memory cell currents for a desired computing precision [164]. The main difference with prior work is that the noise is exploited for stochastic functionality in the proposed operation. Specifically, two types of noise sources are considered: intrinsic noise to each memory cell and externally added noise to each output, e.g., from additional fixed-biased memory cells or using the input-referred current noise of peripheral circuits.

To analyze the stochastic operation, let us consider normally distributed independent noise sources. This assumption is justified due to the dominant white (thermal and/or shot) intrinsic noise for the most practical >100 MHz bandwidth operation, which would be realistic for both floating -gate transistor and memristor-based analog circuits in which memory arrays are tightly integrated with peripheral circuits [16,130]. The current I is sampled and latched at the peripheral neuron, which consists of a current-mode sensing circuitry feeding, in the case of discrete-time networks, a digital flip flop (Fig. 48a). The flip-flop effectively implements a step function of the sampled value so that the probability of latching a digital "1" state is

$$\Pr(V = V_{1'}) = \frac{1}{2} + \frac{1}{2}\mathrm{erf}\frac{\langle I \rangle}{\sqrt{2}\sigma}, \tag{2.3}$$

where σ is the standard deviation of the output current. There are two distinct regimes for the stochastic operation defined by Equation 2.3. If thermal noise dominates, the fluctuations of an output current would be independent of its average value. In this case, Eq. 2.3 matches almost exactly the sigmoid function of Eq. 2.2, given that temperature is inversely proportional to a peak signal-to-noise ratio I_{max}/σ as

$$T = \frac{\sqrt{2\pi}\sigma}{4I_{\max}}, \tag{2.4}$$

With predominant shot noise behavior, $\sigma \propto \sqrt{\langle I \rangle}$. Even in this case, Eq. 2.3 could closely approximate Eq. 2.2 assuming some effective temperature—see Supplementary Information S1 (and its discussion) in Ref. [8] for more details.

The first regime is representative of intrinsic thermal noise in metal-oxide memristors. Indeed, shot noise in such devices would be negligible due to typically diffusive electron transport [169] and relatively small $V_{'1'}$, which should be not much larger than a thermal voltage at room temperature to avoid disturbance of memory state. Note that intrinsic thermal noise is independent of the applied voltage and will be contributed by all memristors in the column, even zero-biased crosspoint devices, thus excluding any input dependence (Fig. 48b). On the other hand, the intrinsic shot noise is characteristic of ballistic transport in nanoscale floating-gate transistors with sub-10-nm channels [170,71]. This noise can be completely cut off by opening the cell's switch (Fig. 48c). For both implementations, the effective computing temperature can be dynamically varied by changing $I_{\max}$. Moreover, the scaling constant can be uniquely selected for each array's input by adjusting its voltage amplitudes – see, e.g., amplifiers marked with 'x' in Fig. 48e. Stochastic dot-product operation and runtime temperature scaling are demonstrated next in the context of two applications.

3.2.2. A Memristor-based Restricted Boltzmann Machine

Our first experiment focused on the demonstration of a restricted Boltzmann machine using 20×20 crossbar circuits with passively integrated Pt/Al_2O_3/TiO_{2-x}/Pt memristors (Fig. 49), fabricated using the device technology reported in Ref. [17]. The integrated memristors are sufficiently uniform for programming with less than 5% tuning error and have negligible

conductance drift over time. Limiting the applied voltage bias across memristors to $|V_{1'}| \leq$ 100 mV prevents disturbance of memory states during the network operation. At such small voltages, the memristor *I-V* characteristics are fairly linear, with $2I(V_{1'})/I(V_{1'}/2) < 1.02$ for all conductive states [17]. (More details on the memristor technology and crossbar circuit operation are provided in the Methods section.)

The studied bidirectional network consists of 10 visible and 8 hidden neurons (Fig. 49a) with synaptic weights implemented as differential memristor pairs. Each visible neuron is connected to a single vertical electrode of the crossbar, while each differential hidden neuron is attached to two horizontal crossbar electrodes (Fig. 48d). The forward propagation of the information, i.e., from visible neurons to hidden ones, and differential sensing is performed similarly to previous work [17]. In the backward pass, digital '1' input from the hidden neuron is implemented by applying $\pm V_{1'}$ to the corresponding differential pair of lines while grounding both lines for zero input. The current is then sensed at the single-line input of the visible neuron. For simplicity, we study the network with zero bias weights. The remaining weights were chosen by first generating random real numbers within the [-1, +1] range. These values were mapped to -32 μS to +32 μS at 50 mV maximum conductance range of a differential pair using the 20 μS conductance bias and the 4–to-36 μS dynamic range of individual devices. After such individual device conductances had been determined, memristors were programmed using automated tuning algorithms [172] with a 5% tuning accuracy to the desired states (inset of Fig. 49).

Figure 49c shows stochastic dot-product results when utilizing external white noise with a fixed standard deviation (inset of Fig. 49c), injected directly in the hardware from the readout circuitry. Specifically, these results were obtained by applying all possible digital inputs to

the hidden neuron #2, and collecting 100,000 samples of the crossbar array output currents for each specific input, while emulating the peripheral circuitry in software. The effective computing temperature, i.e., the slope of the sigmoid function, is controlled by V_{-1}.

In our main RMB experiment, we first apply randomly generated digital voltages to the vertical crossbar lines connected to visible neurons, then sample output currents on the horizontal crossbar lines feeding hidden neurons, and convert sampled values to the new digital voltages of hidden neurons according to the signs of the corresponding differential currents. Note that only functionality of a sensing circuit and latch (i.e., applying step function to the sensed currents and holding the resulting digital value) is realized in software, while the probability function of the Eq. 2.3 is implemented directly in the hardware. In the next step, the calculated voltages at the hidden neurons are applied to horizontal lines, and the new voltages at the input neurons are computed similarly to the forward pass. The voltages at the input and hidden neurons represent the new state of the network after one forward/backward state update ("epoch") and are used to calculate its energy according to Eq. 2.1. These updates are repeated multiple times in a single run of the experiment.

Figure 49d shows the results of this experiment, namely the energy distributions at different effective computing temperatures calculated from Eq. 2.4. Each distribution corresponds to the measured energies in the final 500 epochs of a single run (see an example for such run in Fig. 49e), averaged across 100 different trials that start with randomly chosen initial neuron states. For a wide range of effective computing temperatures, the experimentally measured data are in good agreement with simulations, which were based on the stochastic binary neuron with the ideal sigmoid probability function. Note that because of bipartite

network topology, the system quickly converges to thermal equilibrium, which is indirectly confirmed by comparing energy distributions over different periods (inset of Fig. 49d).

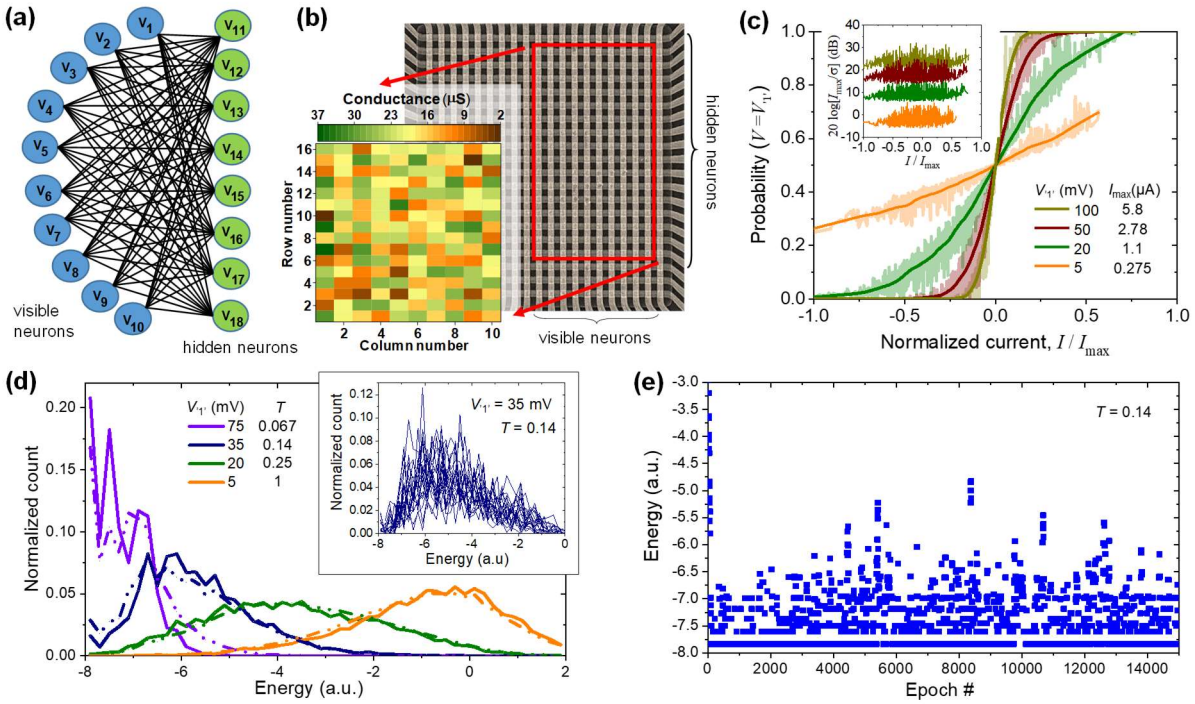


Fig. 49: Memristor-based restricted Boltzmann machine. (a) A bipartite graph of the considered RBM network and (b) its implementation with passively-integrated metal-oxide memristors. The red rectangle highlights the utilized area of the 20×20 crossbar array, while the inset shows the conductance map, measured at 50 mV, after programming devices to the desired states. (c) Measured probability distribution functions at several $V_{1'}$, i.e., different effective computing temperatures, for the hidden neuron #2 of the implemented network, which is attached to rows #3 and #4 of the crossbar circuit. The inset shows peak signal-to-noise ratios across a full range of neuron's input currents. I_{max} corresponds to the largest input current to neuron #2. Some minor, unwanted SNR dependence on the input current is due to the artifacts of the experimental setup. (d) Measured (solid) and modeled (dash-dot) energy distributions. The inset shows measured energy distributions for the specific temperature, collected over 20 different 1000-epoch spans. (e) An example of the simulated evolution of energy for the specific temperature. All neurons are initialized to a zero state at the beginning of this simulation. In all experiments, the neuron's input currents were sampled at 1 MHz bandwidth while integrating noise above this frequency. On panels d and e, the temperature is computed relative to the largest possible current, which corresponds to all ten differential synaptic weights set to the maximum value of 32 μS.

3.2.3. Neurooptimization Based on Floating-gate Memories

In our second experiment, we investigated the implementation of a generalized Hopfield network with embedded NOR flash memory for solving a combinatorial optimization problem

(Fig. 50). The experimental work was performed on a 6×10 integrated array of supercells (Fig. 50a), using 55-nm technology modified from the commercial process for analog tuning [40]. One supercell hosts two floating-gate transistors sharing a common source terminal so that there are 120 memory cells in such array. The subthreshold currents of crosspoint transistors can be tuned uniquely and precisely for each cell within a very wide dynamic range by adjusting the charges at the floating gates, enabling very efficient implementation of dot-product operation in which inputs are applied to word gate (WG) lines and output currents are sensed from the drain (D) lines [130]. Furthermore, the currents can be simultaneously scaled (and even completely suppressed), without re-tuning, for all cells sharing the same coupling gate (CG) / WG line, by controlling CG and/or WG voltage amplitudes while keeping other cell's terminals biased at typical reading conditions. More details on the utilized embedded NOR flash memory devices and circuits are provided in the Methods section.

Figure 50b shows the results of stochastic dot-product operation for the flash memory implementation. For these measurements, currents of 10 cells, sharing a drain line of the memory array, were set with 20% tuning precision to 175 nA, which is a representative value for the considered experiment. After that, 20,000 samples of single-ended bit-line currents were collected at 10 KHz bandwidth for 30 randomly selected inputs. Similar to the RBM study, fixed white noise was added externally directly from the readout circuit (inset of Fig. 50b), while other peripheral functions were emulated in the software. To consider different neuron's input currents, randomly chose *m* cells (out of 10 total) on the bit line, i.e., a specific voltage was applied to the selected *m* WG lines, while the remaining cells were disabled by grounding their WG lines. This experiment was repeated 3 times for each *m* from 1 to 10. The effective computing temperature was controlled by adjusting CG voltage.

Our specific focus is on solving a graph partitioning problem with parameters specified in Fig. 50c. Let us consider a graph (U, E) with N nodes, node weights w_i, and edge weights e_{ij}. Since each node will be uniquely mapped to the corresponding neuron, U_i is also used to define the state of i^{th} neuron. The problem is to partition the graph into two partitions of nearly equal weight such that the cutsize, the number of edges with an endpoint in each partition, is minimized. To solve this problem, let us consider a discrete-time discrete-state recurrent neural network. The intuitive energy function is given by

$$E = \alpha \sum_{i=1}^{n} \sum_{j=1}^{n} e_{ij}(U_i + U_j - 2U_iU_j) + \sum_{i=1}^{n} \sum_{j=1}^{n} w_iw_j(1 - U_i - U_j + 2U_iU_j),$$

where the first term minimizes the weighted sum of edges that belong to the cut, and the second term will have a minimum value when the sum of node weights assigned to the two partitions are equal, while $\alpha = 0.5$ is a constant representing the relative importance of these two terms [173]. By dropping constant terms and rearranging this expression, a more convenient energy function for neural network, for which diagonal weights should be zero to ensure that energy is decreasing during state updates, is

$$E = -\frac{1}{2}\sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} (2e_{ij} - 4w_iw_j)U_iU_j - \sum_{i=1}^{n} U_i \left(2w_i \sum_{j=1}^{n} w_j - 2{w_i}^2 - \sum_{j=1}^{n} e_{ij} \right).$$

(Note that there is a mistake in Ref. [173] in that $-2w_i^2$ term in bias weights is missing.) This equation directly defines neural network array and bias weights:

$$T_{ij} = 2e_{ij} - 4w_iw_j, \quad {T_i}^{\mathrm{b}} = 2w_i \sum_{j=1}^{n} w_j - 2{w_i}^2 - \sum_{j=1}^{n} e_{ij},$$

The corresponding utilized synaptic weights in the floating-gate implementation are

$$I_{\mathrm{cell}} = \begin{pmatrix} 0 & -140 & -478 & -500 \\ -140 & 0 & -508 & -552 \\ -478 & -508 & 0 & -622 \\ -500 & -552 & -622 & 0 \end{pmatrix} \times \frac{(I_{\mathrm{cell}})_{\max}}{622}, \quad {I_{\mathrm{cell}}}^{\mathrm{b}} = \begin{pmatrix} 559 \\ 600 \\ 804 \\ 837 \end{pmatrix} \times \frac{(I_{\mathrm{cell}})_{\max}}{837}$$

where $(I_{cell})_{max}$ is the largest cell current, which is controlled by adjusting WG and/or CG line voltages. Note that bias and array weights are always of different signs for the graph partitioning problem and, these two groups of weights are normalized differently to increase dynamic range and improve nonlinearity. This can be readily implemented in hardware by having different gains for the positive and negative pre-amplifiers in the differential sensing circuitry.

The neural network weights were mapped to the cell currents using $(I_{cell})_{max}$ = 1.0 μA, which resulted in comparable to memristor study range of SNRs. To demonstrate the versatility of the proposed circuits, four different variations of Hopfield networks were considered for solving this combinatorial optimization problem: an original approach (labeled as "baseline") [155]; a scheme with dynamically adjusted problem/energy function ("adjustable"); a network with chaotic annealing ("chaotic") [50]; and, finally, a generalized Hopfield network with simulated annealing implementation, which is enabled by stochastic dot-product circuits ("stochastic"). The implemented network is discrete-time/state with state updates performed sequentially for randomly selected neurons during operation for all approaches.

More specifically, the proposed "adjustable" approach draws inspiration from the work on quantum annealing [174], in which an initial problem is modified to ease convergence to a global optimum. Similarly, we modified the problem by adding an additional node with relatively large weight and zero-weight edges (Fig. 50c). The additional node weight was exponentially decreased from 50 to 0.2 at each update, thus gradually turning the mapped problem and its energy function to those of the original one. In the hardware implementation, the extra node was realized by extending the original memory cell array by one column and

one row (highlighted with green background in Fig. 48e) and decreasing WG voltage from 1.2 V to 0.2 V. Note that the additional bias line was required to separate contribution of bias currents from the original node weights and that of the added one.

For the chaotic annealing approach, we followed the idea of Ref. [50] to utilize transient chaos for better convergence. The chaotic behavior was facilitated by initially employing large negative diagonal synaptic weights (I_{cell} = -1.2 μA at V_{CG} = 1.5 V and V_{WG} = 1.2 V), which were encoded in a separate array of cells (Fig. 48e). These weights were decreased linearly to ~ 0 with each update by changing WG voltage on these additional cells during runtime from 1.2 V to 0 V.

In the baseline, adjustable, and chaotic annealing experiments, all updates were deterministic (i.e., with larger SNR for neuron input currents) due to using larger CG voltages (Table S1), and also very low (~20 Hz) operational bandwidth, which further reduced noise impact. For the stochastic Hopfield network, the nodes were updated probabilistically at 20 KHz bandwidth according to Eq. 2.2. To implement simulated annealing, we exponentially increased CG voltage from 1 V to 2 V in 80 steps, which corresponds to an 80× decrease in effective computing temperature.

Figure 50e shows the main results of the neurooptimization study. The convergence for the baseline approach is fast, though the network often gets stuck in the local minima. As a result, the final energy, averaged over many runs, is substantially higher than the global optimum ("ground state" line on Fig. 50e), which corresponds to the solution shown with a dashed red line in Fig. 50c. On the other hand, an optimal solution was almost always found using the three remaining approaches. For the adjustable approach, the initial increase in energy of the original 4-node problem is expected, given the quick convergence to the global

energy optimum of the 5-node problem. As the additional node is gradually eliminated from the network, 4-node problem energy quickly drops to below baseline level, resulting in a better solution. This is likely due to the network state being very close to its global minima during convergence or a more optimal initial state corresponding to the optimal solution of the 5-node problem. For all considered approaches, experimental data follow simulation results very closely (Fig. 50e). Furthermore, the SPICE simulations at 100 MHz operation bandwidth also show similar performance when using only intrinsic cell noise (see Fig. S3a in the supplementary of Ref. [8]).

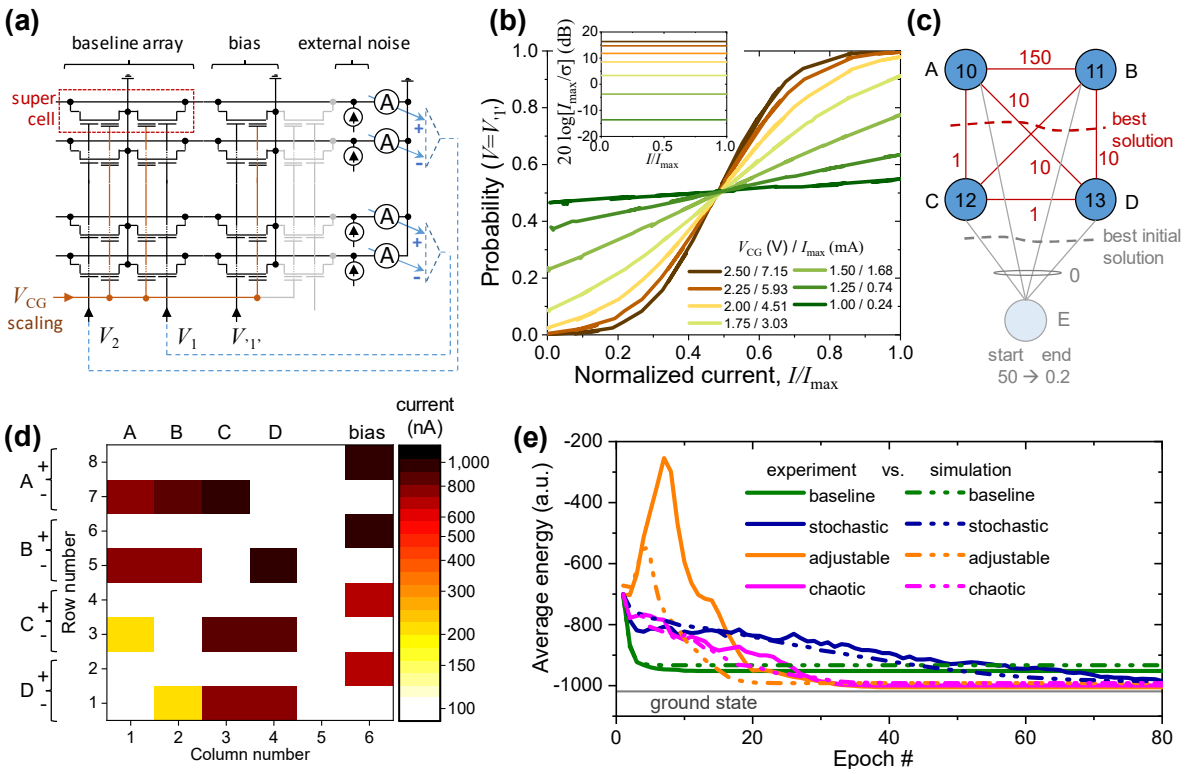


Fig. 50: Neurooptimization based on floating-gate memory arrays. (a) Schematics of the experiment, shown for clarity for the 2×2 supercell array implementing a baseline/stochastic network with two neurons. The parts shown with dashed lines were emulated in software. (b) Measured probability distribution functions for single-ended stochastic neurons at various CG voltages (or effective computing temperatures). The inset shows measured peak SNR across the full range of neuron input currents. I_{max} is the largest measured input current to the neuron. (c) Implemented graph partitioning problem with considered edge and node weights. The wavy lines show the cuts corresponding to the best solution. Shaded nodes/edges are used for the method with dynamically adjusted energy functions. (d) Conductance map for the main

section (highlighted with a blue background in Fig. 47e) of the weights in the experiment, after tuning with 3% precision. For the fixed synaptic weights, the tuning was performed at the operating biases. For the variable weights, the tuning was performed at the lowest (largest) CG and WG voltages during operation for the stochastic (adjustable and chaotic) approaches. (e) Simulation and experimental results for neurooptimization. For the first three (chaotic) cases, the data are averaged over 100 (20) runs for each of the initial states, with a total of 1600 (320) runs. The energy function for the "adjustable" case is calculated by taking into account only four original nodes. The biasing conditions during operation are summarized in Table S1.

3.2.4. Device and Circuit Nonidealities

As discussed in the main text, there are two specific issues for the floating-gate memory implementation, which might impact functional performance. Specifically, an approach of using differential summation and intrinsic cell noise and the problem of variations in the subthreshold slope can both lead to deviation of the neuron's probability density function from the ideal one. Such variations can be treated as "smearing" of the effective computing temperature. More specifically, the first problem can be illustrated by considering two extreme cases, namely when subtracting two smaller similar currents and two larger similar currents on the differential lines. The total differential current could be comparable, though due to the dependence of the intrinsic shot noise on the cell currents, the signal-to-noise ratio would be larger (and hence temperature smaller) for the latter case. We considered a 100-node graph partitioning problem with randomly distributed weights and edges within [0,1] interval to investigate this issue. Figure 51 shows the corresponding neural network weight map, in which the array and bias weights are separately normalized with respect to their largest values. The probability density function was then simulated by adding shot-like noise $\sigma^2 = \alpha I$ to differential lines and considering different combinations of input currents for all neurons. The overall smearing is rather negligible, more so at lower temperatures (Fig. 51b).

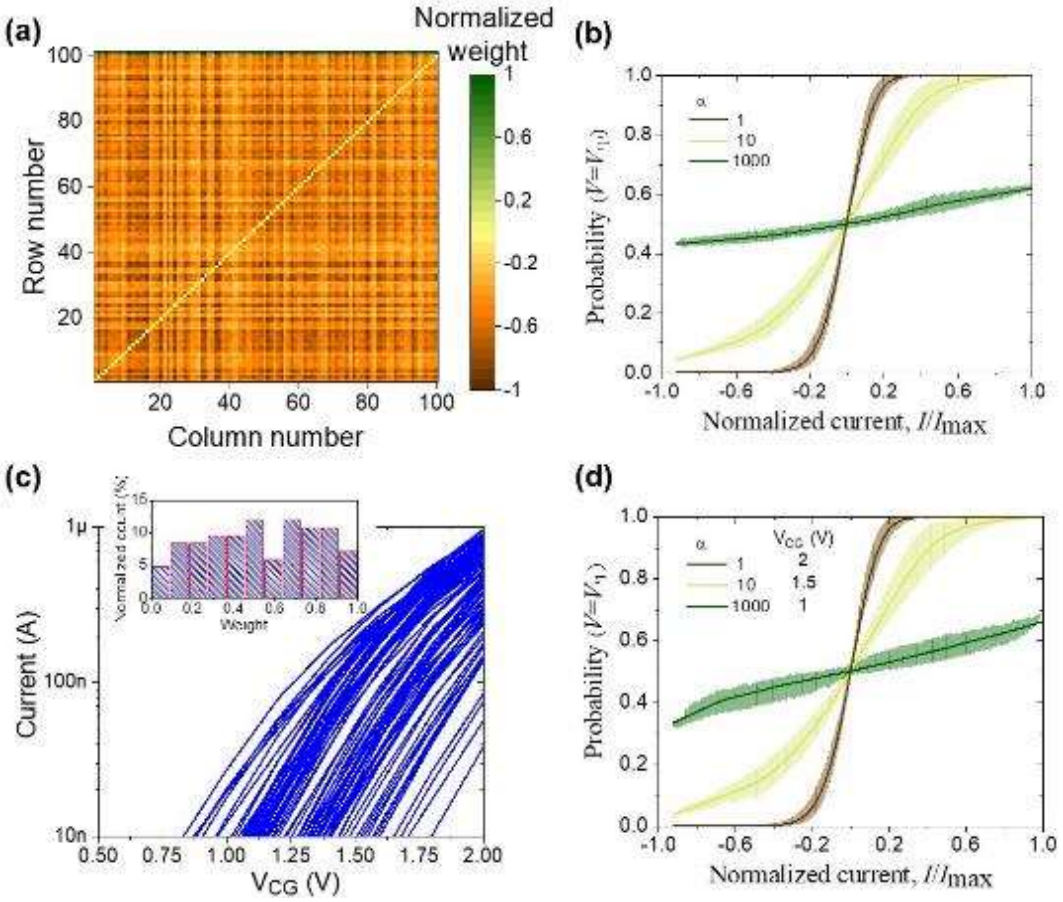


Fig. 51: Non-idealities in the stochastic dot-product circuit based on floating-gate memory: (a) The considered distribution of neural network weights, normalized separately for positive (array) and negative (bias) weights. The bias weights are shown at the very top of the array. (b) Temperature smearing of the stochastic neuron function due to input-dependent output-referred current noise in differential circuits for three effective computing temperatures. (c) Measured subthreshold slope *I-V*s for 100 memory cells. The inset shows the histogram of corresponding normalized synaptic weights, defined as $I_{cell}(V_{CG}=2.0\ V)/(I_{cell})_{max}$, when using $(I_{cell})_{max}$ = 973 nA. (d) Simulated combined temperature smearing due to the subthreshold slope variations and differential summation in floating-gate implementation.

The second problem arises from the nonlinearities in the subthreshold *I-V* characteristics of flash memories. Ideally, the relative cell currents (and hence synaptic weights) should scale with the same rate when changing CG voltage for the considered annealing schemes. In practice, however, these currents scale differently due to process-induced variations and voltage-dependency of the subthreshold slope. To quantify this issue, we have measured subthreshold characteristics of the 100 devices, which were tuned randomly at V_{CG} = 2 V, V_{WG} = 1.2 V) to currents ranging from 40 nA to 1 μA (Fig. 51c), and then used these data to

simulate the resulting effective temperature smearing due to both differential summation and slope variations for the aforementioned graph partitioning problem. As Figure 51d shows, the additional smearing is rather negligible at all considered effective computing temperatures.

Due to non-negligible tuning error in conductance and its drift over time, the effective precision of weights in mixed-signal circuits with analog-grade nonvolatile memories is typically limited to below 8 bits. Such precision is adequate for the inference operation of many deep-learning feed-forward and recurrent models. According to our preliminary simulation results (Fig. 52), it is also sufficient for the studied applications, though this issue has to be certainly further studied for larger, more practical networks.

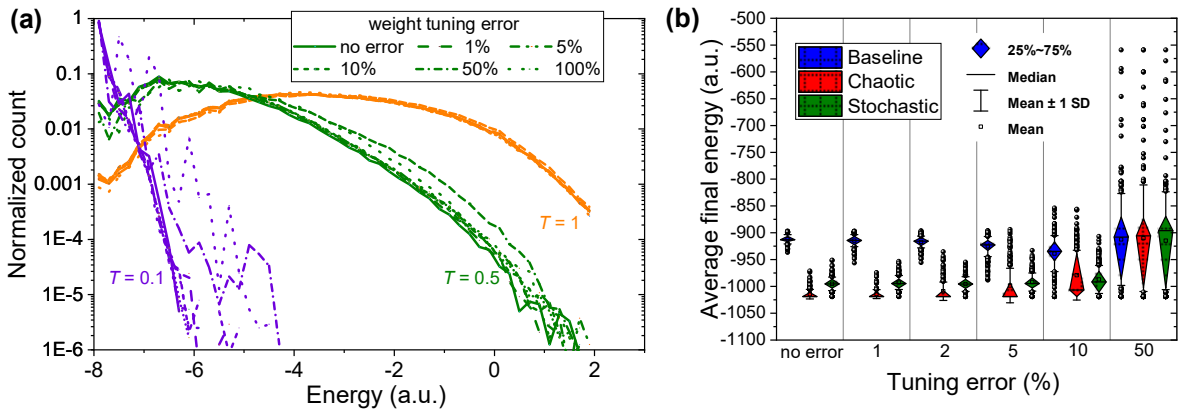


Fig. 52: The impact of weight precision on functional performance. The simulation results for (a) RBM energy distribution and (b) energies after the 80th epoch of graph partitioning problem for the same networks considered in the main text, obtained with different assumptions for the conductance tuning. For RBM simulations, the data were collected over 500 epochs and were averaged over 500 different trials. In the neurooptimization experiment, 200 sets of weights were generated for each case of tuning error. A single data point on a graph represents energy achieved after 80th averaged over 16×10 trials (10 runs for each of the 16 initial states) for a specific set of weights. For clarity, data points inside 25%-75% are not shown. The tuning error was simulated by randomly choosing weights from the range of target value × [1 - tuning error, 1 + tuning error]. The energy is calculated assuming target (error-free) weights in both panels to make the comparison meaningful.

3.2.5. Methods

The restricted Boltzmann machine is implemented with a 20×20 passively integrated ("0T-1R") memristive crossbar circuits fabricated in UCSB's nanofabrication facility. The active bilayer was deposited by low temperature reactive sputtering method, while crossbar electrodes were evaporated using oblique angle physical vapor deposition and patterned by lift-off technique using lithographical masks with 200-nm lines separated by 400-nm gaps. Crossbar electrodes are contacted to a thicker (Ni/Cr/Au 400 nm) metal line/bonding pad, which was implemented at the last step of the fabrication process. Similar to Ref. 17, the majority of the devices were electroformed, one at a time, by applying one-time increasing amplitude voltage sweeps using an automated setup. Automated "write-verify" tuning algorithm [114], involving the alternative application of write and read pulses, was used for setting the memristor conductances to the desired values. Specifically, the memristors were formed/written one at a time using the "V/2-biasing scheme," i.e., by applying half of the write voltages of the opposite polarity to the corresponding two lines connected to the device in question while floating/grounding the remaining crossbar lines.

The formed memristors have fairly uniform switching characteristics, with set and reset voltages varying within 0.6 to 1.5 V and -0.6 to -1.7 V, respectively. The memristors' *I-V*s are nonlinear at larger biases due to the aluminum oxide tunnel barrier in the device stack, which helps with limiting leakage currents via half-selected devices during programming. The sneak path currents during network operation and read phase of tuning algorithm are negligible because all lines are always tied to some voltages and also very large conductance of lines (~1 mS) compared to those of the crosspoint memristors (<36 μS), allowing for < ~5% computing

error. More details on fabrication, electrical characterization, and memristor array operation can be found in Refs. 25 and 34.

The 12×10 arrays of floating gate cells were fabricated in a commercial 55-nm embedded NOR memory process, redesigned for analog applications. The array matrix is based on "supercells", which consist of two floating-gate transistors sharing the source (S) and the erase gate (EG) and controlled by different word (WG) and coupling (CG) gates. The cells are tuned using the write-verify tuning procedure. (Note that WG, D, and S supercell terminals are typically denoted by, respectively, WL, BL, and SL in the context of digital memory circuits. The new labels are more relevant to the considered application and were used to avoid possible confusion.) After the weight tuning process had been completed, the network operation was performed using V_D =1 V, V_{WG}= 0.8 V, V_S= 0 V, V_{EG} = 0V, and $V_{CG} \in$ [1 V, 2V]. Such biasing conditions were mainly imposed by the requirement of keeping the transistors in subthreshold region, ensuring large (>30 dB) dynamic range of signal-to-noise ratios, and minimizing the impact of subthreshold slope variations on weight scaling – see Section S7 of the Supplementary Information of Ref. [8]. The details of the characterization setup used for both memristors and flash memories are also provided in Section S6 and S7 of the Supplementary Information of Ref. [8].

3.2.6. Discussion

The considered case studies allow contrasting stochastic dot-product implementations with two representative memory technologies. The main advantage of floating-gate memory devices is their mature fabrication technology, which can be readily used for implementing practically useful, larger-scale circuits. Their substantial drawbacks for the considered applications include unipolar electron transport, which, e.g., necessitates using two sets of

cells with similarly tuned conductances for the forward and backward computations in RBM networks. Furthermore, for differential current sensing, the total injected noise depends on the absolute currents on the differential lines rather than their subtracted value. Due to variations in subthreshold slopes, there are also noticeable changes in relative weights when scaling currents (Fig. 51c). Fortunately, the resulting smearing of the probability distribution function due to these issues is rather weak, which is confirmed by extensive modeling results in Fig. 51. (Note that due to the input-independent noise and linear device *I-V* characteristics of at small biases, there are no such problems for memristors.) Finally, floating-gate cells are also sparser and less scalable. However, these deficiencies are somewhat compensated by lower peripheral overhead due to the cells' high input and output impedances [130], and also by having more design options in scaling cell currents due to multi-terminal cell structure, which is important for the considered annealing approaches.

On the other hand, metal-oxide memristors are arguably the most prospective candidate for the proposed circuits. Their major challenge is immature technology requiring substantial improvements in device yield and *I-V* uniformity. The improved device technology should also feature lower cell currents, by approximately two orders of magnitude, to improve system-level performance and to allow for highly effective temperatures during operation when relying on intrinsic noise in the stochastic dot-product circuits. Due to the linear dependence of the off-state current on the device footprint, cell currents in the utilized memristor technology can be reduced by scaling-down device features [17]. Moreover, memristors with a suitable range of resistances based on other materials have also recently been reported [67, 68], and further progress in this direction can be helped by the development of foundry-compatible active metal-oxide memristor (1T-1R RRAM) macros [172].

It is worth noting that similar to other applications [16], a limited tuning precision and switching endurance for memristors and flash memories should be adequate for "inference"-like computations in both studied applications. For example, simulation results in Fig. 52 show almost no degradation in performance for up to ~5% and ~10% tuning errors (which is crudely equivalent to 3 and 2 bits of weight precision) for the considered RBM network and graph partitioning problem, respectively. We also envision that the proposed neuro-optimization hardware will be the most useful for computationally–intensive problems and thus require relatively infrequent weight re-tuning because of longer runtimes. In principle, implementations based on high-endurance digital memories, such as ferroelectric devices [175], would broaden the application space for the proposed circuits, e.g., enabling RBM training. Such implementations, however, would require multiple digital devices per synaptic weight, resulting in sparser designs with worse performance and energy-efficiency.

2.12. Weight Annealing in Memristive Hopfield Networks

Following the previous section, we introduce a compact and scalable weight annealing technique of the Hopfield model with its VLSI design, which is more effective and more scalable than conventional annealing methods. Our approach dates back to methods like weight annealing [176-179], noising [180], space smoothing [181], and fine-tuned learning [182], where the core idea is to change in the energy landscape by modifying weights in the formula for the energy. Here, the exact meaning of "weight" varies from method to method, as well as from problem to problem addressed – a weight may be associated with an input data point, a subproblem, etc.; similarly, a variety of ways to modify the weights (random perturbation, adversarial change, etc.) has been explored. The common crux of the methods is that they modify the weights differently in every timestep and in different areas of the solution

space; this way, the search is guided by weight changes adapted to the current state and reuses insights gained from previous iterations. While the clever schemes for such adaptive weight modifications underpin the strengths of the methods, mimicking this adaptivity within any hardware would likely be inefficient since performing individual changes to the weights consumes significant time and energy. Further, hardware implementation of the algorithms which act differently in different parts of the solution space would require a complicated circuitry, leading, again, to efficiency losses.

Our proposed weight annealing circumvents both of the above: First, at any iteration, all weights are scaled together. Second, the weight modification is oblivious to the status of the solution space exploration – the annealing schedule is pre-set in advance and does not depend on the state of the system (in particular, the schedule does not depend on the value of the energy function – it is the hardware that takes care of the derivatives, convergence, escaping local optima with stochastic decisions, etc.). We numerically demonstrate the effectiveness of our approach on several benchmarks by solving graph partitioning, vertex cover, maximum-weight independent set, and maximum-weight clique problems.

3.3.1. Hardware-friendly Weight Annealing

The Lyapunov energy associated with a certain state of the Hopfield network at t is given by

$$E(t) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(t)U_i(t)U_j(t) - \sum_{i=1}^{N} T_j^{\mathrm{b}}U_j(t)\,. \tag{3.3.1}$$

where the parameters have their usual meaning. The idea is to slowly modulate the energy landscape, starting from a funnel shape with a deep global optimum where the ground state is easily accessible. The network traps in it in the early stages and tends to remain in the ground

states during the weight modification. In our proposed method, the synaptic weights are slowly changed using $w_{ij} = T_{ij}(1 - e^{\frac{-t}{\tau}})$ where τ>0 is the annealing schedule, and T is the ultimate synaptic weight matrix. At the beginning $(t = 0)$ and while the first term (in Eq. 3.3.1) is zero, the total energy of the network is $-\sum_{i=1}^{N} T_j^b U_j(t)$. It is very straightforward to locate the optimum solution, and the network captures it very fast since there are no local minima. The ground state, for example, is located at $U_j = 1$ for the j^{th} neuron that has $T_j^{\text{b}} > 0$. As the network evolves, $w_{ij}(t)$ gradually moves toward $T_{ij}(t)$ and the first term in Eq. 3.3.1 becomes more significant until the network stabilizes in the equilibrium state. This approach is not only very efficient in terms of solution quality in comparison with other annealing techniques, but also it has a very straightforward implementation with crossbars of the mainstream analog nonvolatile memories. Note that quantum annealing of artificial Ising spins [10] employs a similar concept in which the system Hamiltonian is first fully characterized by a transverse wave and then gradually turned off while increasing the weight of the Ising Hamiltonian.

We consider graph partitioning problems that find applications, e.g., in graph-based electronic structure theory applied to quantum molecular dynamic simulations [183]. A 7-node graph partitioning problem with randomly selected weights and edges is shown in Fig. 53a (see Supplementary Section 3 of Ref. [184] for the actual vertex and edge weights). Fig. 53b shows the semi-exponential energy evolution (of all possible states) during annealing $(\tau = 40)$. The energy associated with each state is exponentially increasing as expected. The black sphere points (projected to the bottom plane for clarity) represent the ground state of the system during the annealing. The global optimum is -389.5459 and locates at state 97 (decimal equivalent of "1100001"). The transitory state is specified by listing the N values of

U_j and represented by a binary word of N bits and its decimal version for simplicity. At $t = 1$, the global minimum is recognizable (state 118, E=−917.76). While the network is steadily evolving, the ground state increases, and its location changes several times. The average transitory energy of the system (defined over the transitory synaptic weights) is also shown for 128 initialization schemes and 200 epochs ($N_{\mathrm{EP}} = 200$) in magenta. The network finds the initial ground state very quickly (regardless of the initial state) owing to the annealing mechanism and tracks it during the evolution.

Fig. 53c shows the performance of the weight annealing technique versus stochastic annealing with a probabilistic sigmoid neuron (the temperature is reduced exponentially from 100 to 0.01) and chaotic annealing (the self-feedback weights are decreased exponentially from 250 to 0.001). In this experiment and after 200 epochs, the success rate (the relative number of cases led to the global optima) is 57.8%, 59.37%, 94.53% for chaotic, stochastic, and weight annealing techniques, respectively, and it is 28.12% for the standard Hopfield model (baseline). It is noteworthy that the stochastic annealed network converges to E=−387.98 and scores a 98.6% success rate when we use 30k epochs and scale the temperature from 100k to 0.01.

To further investigate the proposed approach, 200 randomly populated configurations of 5, 10, 15, 20, and 25-node graphs are considered. The annealing schedule parameter is manually optimized for the first problem and used in all configurations (see Table III for the list of the simulation parameters). The scalability of our approach is compared with simulated annealing on three scenarios: first, we assume a fixed epoch size of 300 for all sizes, then N_{EP} is exponentially increased for 25-node graphs, and then, N_{EP} is exponentially increased with respect to the linear increase of the problem size.

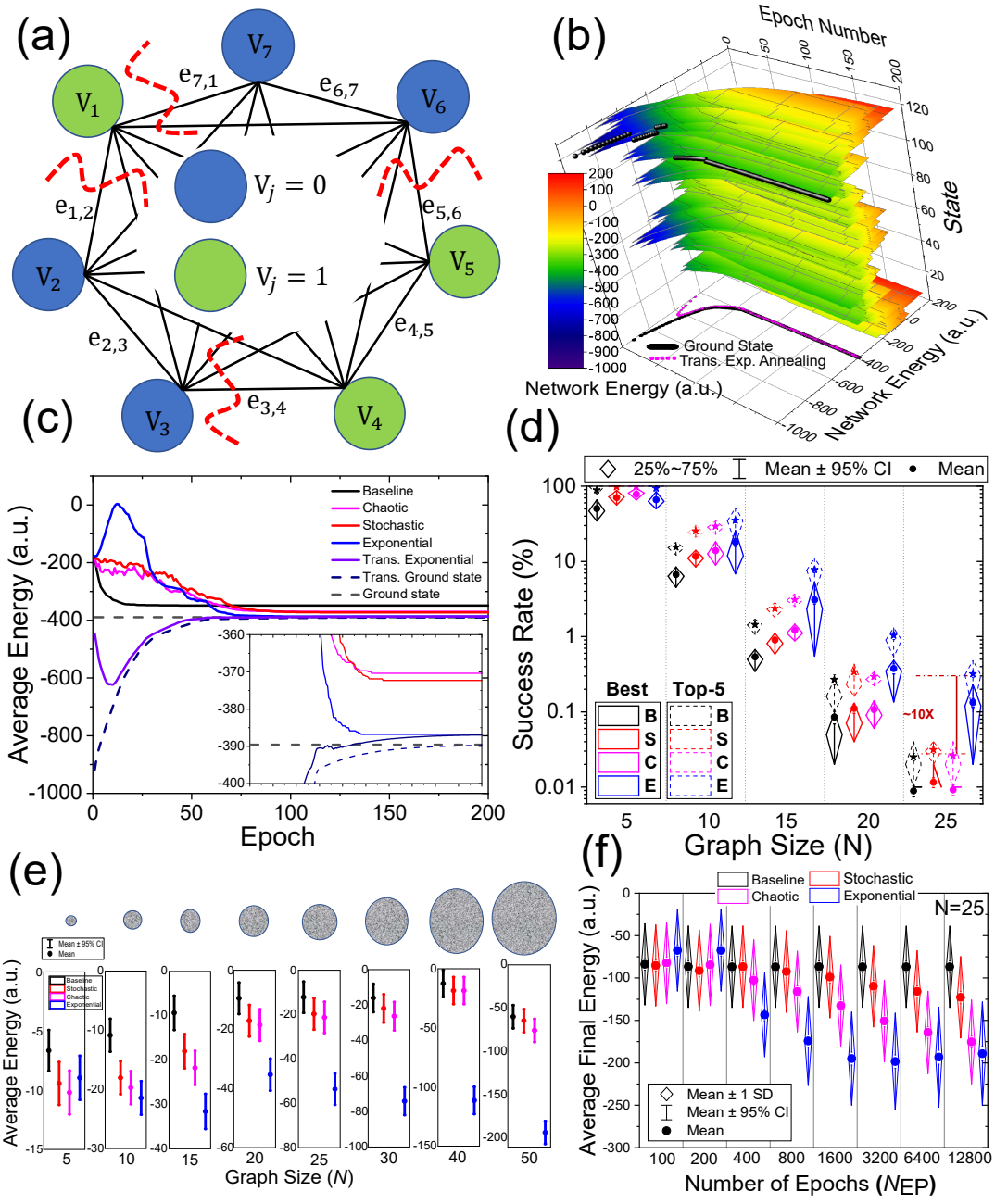


Fig. 53: Neuro-optimization with the weight annealing: (a) The 7-node weighted graph partitioning problem that is used to illustrate the mechanism of weight annealing. The blue/green coloring shows the optimum solution (Supplementary Materials S3 includes the actual weights). (b) The energy evolution of each state during weight annealing for 200 epochs and τ=40. The black spheres mark the transitory ground state of the system, which is also projected to the energy-epoch plane. The magenta curve shows the average transitory energy over 128 runs, which shows that the proposed weight annealing tracks the transitory globally optimum state of the system. (c) The average energy of the network with different annealing techniques over 128 runs. (d) Top-1 and Top-5 success rates of varying annealing techniques versus problem size (B: Baseline, i.e., the standard Hopfield network without annealing, S: Stochastic (temperature reduced from 100 to 0.01), C: Chaotic (temperature reduced from 250

to 0.001), and E: Exponential Weight annealing). For each graph size, we consider 200 randomly weighted problems and provide the parameters in supplementary S4. Note that the best response is the global optimum, and Top-5 counts if the final response is among the best top-5 solutions. Panel (e) shows the distribution of the final average energy, offset by a constant for clarity, for the same graphs used in panel (d). The circles represent graph size. (f) The boxplot of the average final energy vs. epoch size for 200 random configurations of 25-node graph partitioning problems.

We show the success rate achieved by different methods on various problem sizes for $N_{\mathrm{EP}} = 300$ in Fig. 53d. The performance of weight annealing is on par with simulated annealing for N=5; however, the energy gap becomes significantly wider for larger problem sizes. More interestingly, for N=15, among the 200 configurations, the 20 percentiles success rate of weight annealing is better than the 80 percentiles of all other methods. Note that due to the analog capability of our synaptic weights, we consider weighted graph problems, and it would be unfair to compare our results (in terms of success rate) with previous implementations, focusing on sparse graphs with binary weights. Fig. 53e shows the average final energy for the same graphs. We observe that the gap between the solution quality (final energy) of exponential weight annealing and other methods becomes wider in more massive graphs. In Fig. 53f, the computational runtime (epoch number) is increased for 200 configurations of 25-node graphs. We notice that, as expected, the performance of all annealing techniques, including weight annealing, improves by increasing the number of epochs (in part due to slower cooling, which allows the networks to search for better solutions). The performance of weight annealing no longer improves for N_{EP}>3200, while simulated annealing techniques, with noticeable inferior performance, benefit from the longer computational time and slower annealing. This is partly due to the inherent differences between the underlying mechanism of simulated and exponential weight annealing. Stochastic annealing requires more time to explore larger searching spaces. While for the weight annealing, it is simply not the case. The accuracy saturation stems from the fact that slower

learning no longer creates a more optimum path. Note that weight annealing achieves the same solution quality 10× faster than simulated annealing techniques. The simulation results for the case of increasing N_{EP} exponentially with respect to the problem size are shown in Fig. 54. It is observed that weight annealing outperforms simulated annealing for N>10. Similar sets of simulations are also performed for three other combinatorial optimization problems. The trends in the results (Figs. 55-57) indicate that weight annealing outperforms other annealing techniques. For the vertex cover, we notice that the weight annealing fails to find the exact solution (i.e., the global optimum) for few large graph configurations. (Note that weight annealing is still better or on par with simulated annealing on average, outperforms simulated annealing in terms of response quality (Top-5 and average energy.) Spurious states generated during the evolution are responsible for those rare cases.

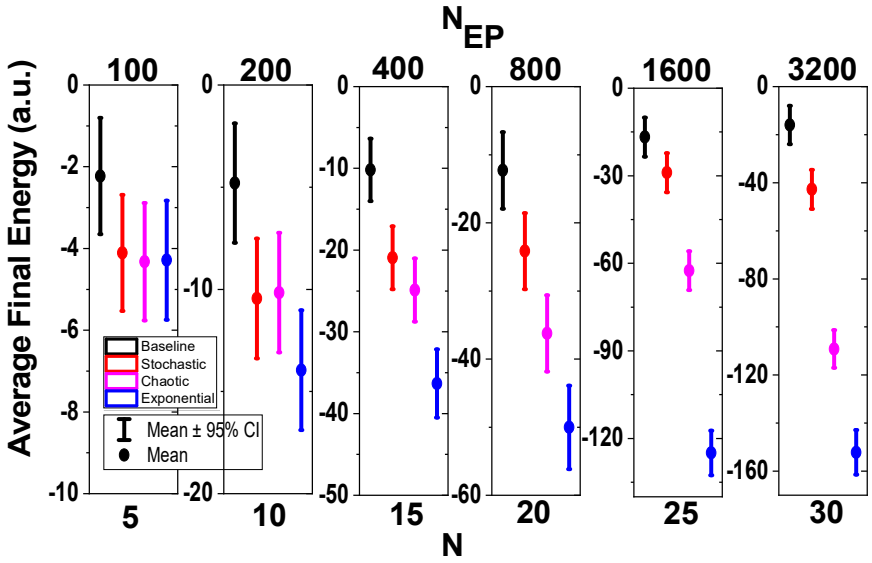


Fig. 54: Extended graph partitioning simulations: The result of increasing the computational time (N_{EP}) with respect to the graph size (N). Note that, in this figure, the average energy is added by a constant for better clarity.

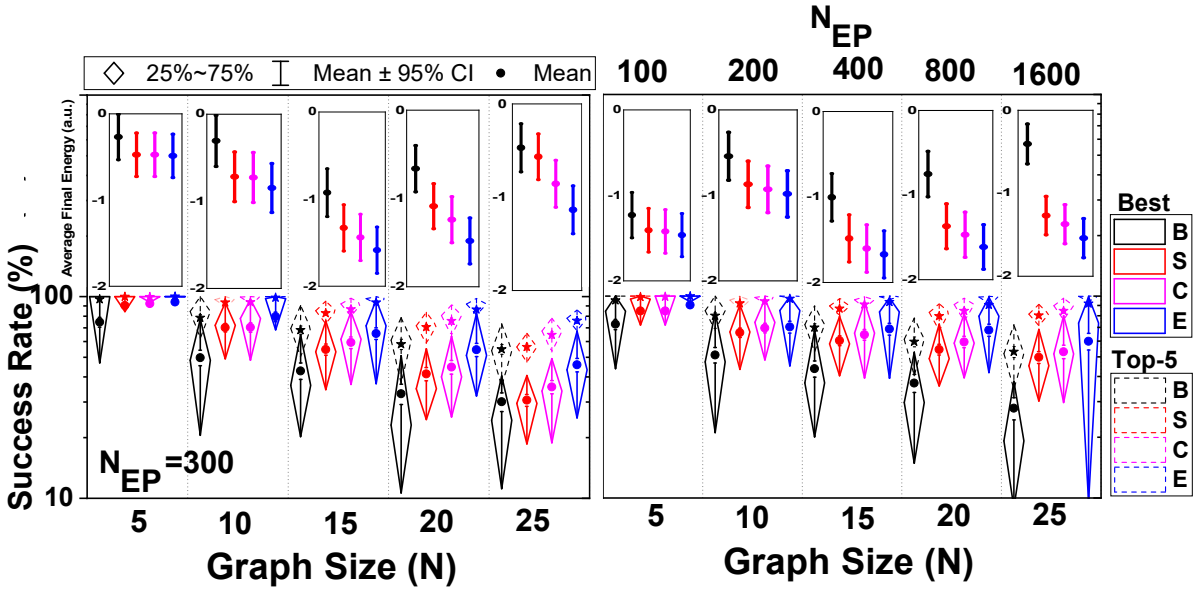


Fig. 55: Simulation results on 200 random configurations of the maximum weighted independent set problem with various sizes: (a) the success rate and (the average final energy in the inset) for different sizes and fixed computational time (N_{EP}=300), (b) the success rate and (the average final energy in the inset) for different sizes with a varying number of epochs. Note that, in each figure, the actual average energy is added by a constant for better clarity.

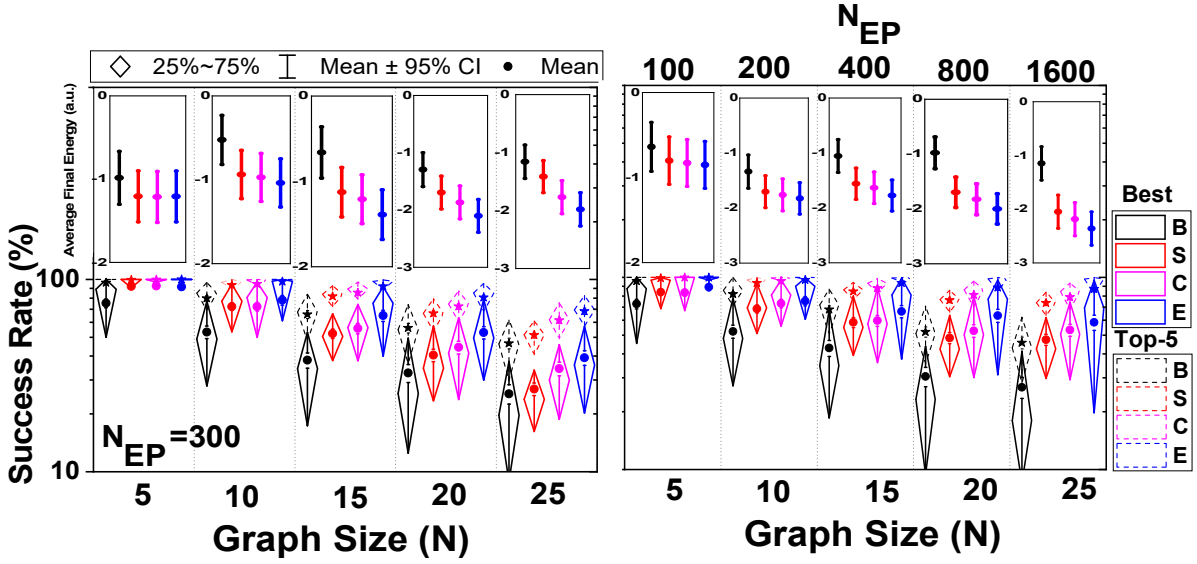


Fig. 56: Simulation results on 200 random configurations of the maximum weighted clique problem with various sizes: (a) the success rate and (the average final energy in the inset) for different sizes and fixed computational time (N_{EP}=300), (b) the success rate and (the average final energy in the inset) for different sizes with a varying number of epochs. Note that, in each figure, the actual average energy is added by a constant for better clarity.

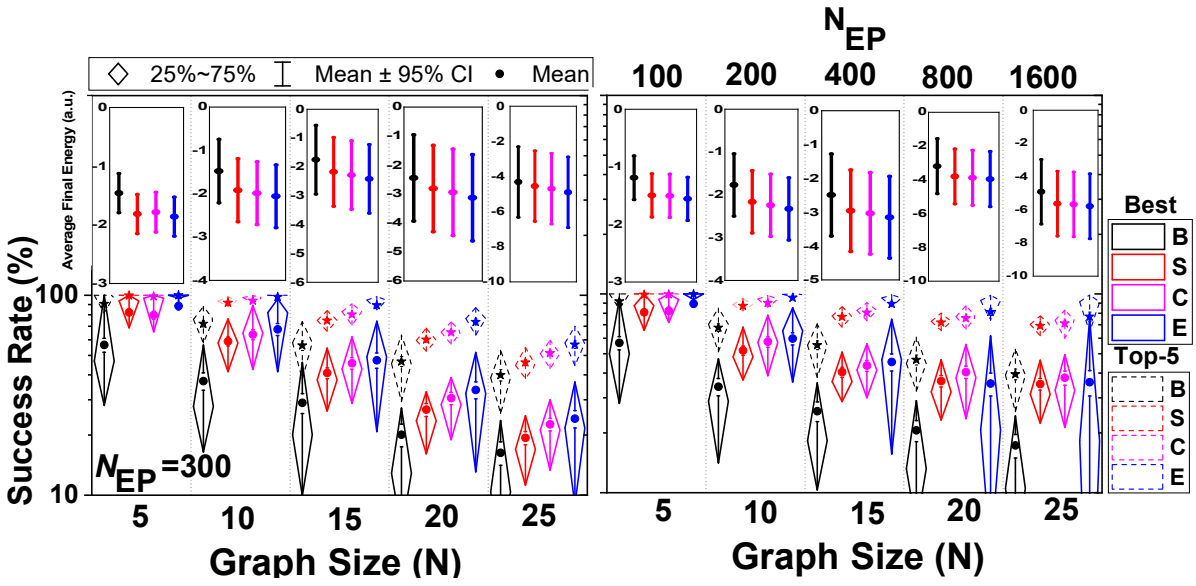


Fig. 57: Simulation results on 200 random configurations of the minimum weighted vertex cover with various sizes: (a) the success rate and (the average final energy in the inset) for different sizes and fixed computational time (N_{EP}=300), (b) the success rate and (the average final energy in the inset) for different sizes with a varying number of epochs. Note that, in each figure, the actual average energy is added by a constant for better clarity.

Table III: The parameters used in the graph-partitioning simulations

	N	Epoch Number	Case numbers	Annealing Schedule		
				Stochastic	Chaotic	Exponential
Fig.52e	5	300	32	50	50	60
	10	300	1000	60	500	60
	15	300	10000	70	1e3	60
	20	300	10000	70	5e3	60
	25	300	10000	100	9e3	60
	30	300	10000	300	2e4	60
	40	300	10000	500	1e5	60
	50	300	10000	1000	2e5	60
Fig.52f	25	100	10000	150	1e3	20
	25	200	10000	200	4e3	40
	25	400	10000	1e3	1e4	80
	25	800	10000	5e3	1e5	160
	25	1600	10000	1e4	1e6	320
	25	3200	10000	5e4	1e7	640
	25	6400	10000	1e5	1e8	1280
	25	12800	10000	1e6	1e9	2560
Fig.53	5	100	32	30	30	20
	10	200	1000	60	60	40
	15	400	10000	90	3e3	80

	20	800	10000	200	7e3	160
	25	1600	10000	700	1e5	320
	30	3200	10000	2e3	1e6	640
	40	6400	10000	1e4	1e7	1280
	50	12800	10000	1e5	1e8	2560

3.3.2. Experimental Results

The proposed technique is demonstrated by addressing two optimization problems based on the most prospective analog-grade memory technologies. The central merit of weight annealing lies in its very straightforward and compact implementation. We present this by realizing a 16-node graph partitioning problem using a 20×20 passively integrated analog-grade memristive crossbar and a 7-node maximum-weighted independent set on a 12×10 embedded array of eFlash memories.

Fig. 58 shows the implementation of the weight annealing technique. The corresponding hardware realization of Eq. 3.3.1 is discussed in the Methods section for both cases. The main challenge in realizing the weight annealing is scaling the synaptic weights. We would like to emphasize that direct modification of (analog) states is impractical in part because of the limited endurance, device-to-device, and cycle-to-cycle variations. This challenge can be resolved in resistive memories by using a simple control circuit (the pre-synaptic drivers), which scales all synaptic weights simultaneously (see Fig. 58). Here, V_{ctrl} is exponentially increased toward V_{ap} at which all devices are tuned. The current neuron state determines which devices should be driven by V_{ctrl}. The post-synaptic circuits include trivial circuits such as transimpedance amplifiers (e.g., a buffered version of Ref. [140]) that senses currents and a dynamic voltage comparator (see, e.g., [185]) that updates the selected neuron state. These circuit functionalities are emulated with Agilent characterization tools.

In split-gate embedded Flash memories, the situation is more straightforward since we can bias the memories in the weak inversion regime, making their states (i.e., currents) semi-exponentially dependent on the select-gate voltage. Then, V_{ctrl} is applied to the shared select-gates and linearly increased toward the V_{ap}.

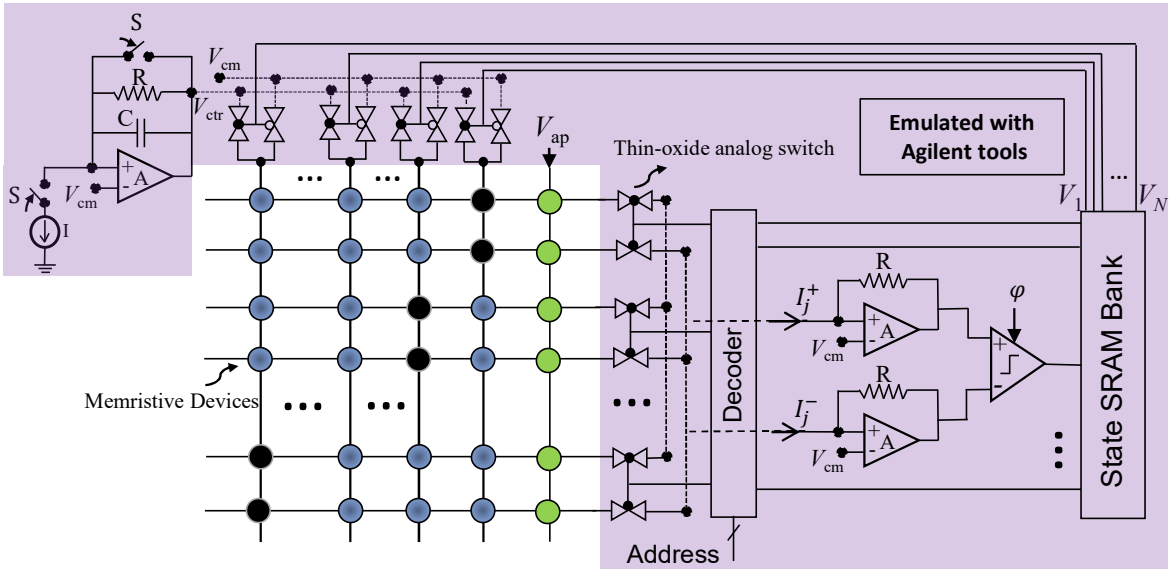


Fig. 58: The current-mode recurrent circuit that implements the weight annealing of discrete-time Hopfield networks with programmable analog memories. The green circles show the bias weights (T_i^{b}) while the black circles implement self-feedback weights (T_{ii}), and the rest of them denote the main synaptic weights ($T_{ij,i\neq j}$). A constant 'on' voltage, which is the same as the tuning voltage, drives the bias column. We control the applied voltage to the rest of the devices during the runtime to adjust the synaptic weights (exponentially). Note that V_{cm} is only added to emphasize that the circuit operate on a single-V_{dd}. Values $R, C,$ and I depend on the problem size and technology and determine the annealing schedule. Switch S resets the network to the initial condition. The selected neuron is determined by the input address to the decoder, and the operation is synchronized with the sampling clock (φ) in the dynamic comparator. Note that we have omitted the tuning circuits in the figure for clarity.

In the first experiment, a 13-node graph partitioning problem is implemented using passively-integrated memristive crossbars. We would like to note that, to the best of our knowledge, this work is the largest Hopfield network circuit implemented with passive memristors. Fig. 59a shows the wire-bonded chip, crossbar TEM image as well as SEM image of the device stack. This crossbar has been previously used for the demonstration of a multilayer perceptron, an integrated spiking neural network for coincident detection, and a

hardware security primitive design. The method section includes a brief description of fabrication steps. Further details are available in our previous works [17,14].

In order to increase the demo size and without the loss of generality, we have ensured the weights and edges (of the graph) are selected such that $T_{ij} < 0$ and $b_j > 0$. This facilitates a single-ended time-multiplexed dot-product of a 13×(13+1) network on our memristive crossbars. A random graph G=(V, E) with the following analog weights are used:

$$V = \begin{pmatrix} 0 & 11 & 6 & 4 & 6 & 1 & 1 & 16 & 18 & 25 & 9 & 4 & 5 \\ 11 & 0 & 12 & 29 & 30 & 2 & 16 & 10 & 14 & 5 & 14 & 9 & 17 \\ 6 & 12 & 0 & 30 & 15 & 3 & 25 & 6 & 18 & 0 & 1 & 5 & 14 \\ 4 & 29 & 30 & 0 & 14 & 8 & 3 & 12 & 8 & 3 & 2 & 19 & 5 \\ 6 & 30 & 15 & 14 & 0 & 7 & 16 & 18 & 7 & 8 & 5 & 13 & 5 \\ 1 & 2 & 3 & 8 & 7 & 0 & 5 & 14 & 1 & 28 & 16 & 30 & 7 \\ 1 & 16 & 25 & 3 & 16 & 5 & 0 & 15 & 14 & 5 & 3 & 4 & 0 \\ 16 & 10 & 6 & 12 & 18 & 14 & 14 & 0 & 5 & 13 & 18 & 20 & 13 \\ 18 & 14 & 18 & 8 & 7 & 1 & 14 & 5 & 0 & 16 & 13 & 19 & 14 \\ 25 & 5 & 0 & 3 & 8 & 28 & 5 & 13 & 16 & 0 & 18 & 18 & 17 \\ 9 & 14 & 1 & 2 & 5 & 16 & 3 & 18 & 13 & 18 & 0 & 10 & 2 \\ 4 & 9 & 5 & 19 & 13 & 30 & 4 & 20 & 19 & 18 & 10 & 0 & 14 \\ 5 & 17 & 14 & 5 & 5 & 7 & 0 & 13 & 14 & 17 & 2 & 14 & 0 \end{pmatrix}, E = \begin{pmatrix} 10 \\ 5 \\ 4 \\ 6 \\ 3 \\ 6 \\ 5 \\ 13 \\ 5 \\ 14 \\ 10 \\ 5 \end{pmatrix}$$

Then, we use the common procedure of finding the weights of the Hopfield network by comparing the general Lyapunov energy with the energy function of the problem (see Supplementary Section 2 of Ref. [184] for more information):

$$T = - \begin{pmatrix} 0 & 178 & 148 & 192 & 228 & 118 & 238 & 168 & 484 & 150 & 542 & 392 & 190 \\ 178 & 0 & 56 & 42 & 42 & 56 & 88 & 80 & 232 & 90 & 252 & 182 & 66 \\ 148 & 56 & 0 & 20 & 20 & 42 & 46 & 68 & 172 & 80 & 222 & 150 & 52 \\ 192 & 42 & 20 & 0 & 92 & 44 & 114 & 76 & 244 & 94 & 276 & 162 & 90 \\ 228 & 60 & 66 & 92 & 0 & 58 & 112 & 84 & 298 & 104 & 326 & 214 & 110 \\ 118 & 56 & 42 & 44 & 58 & 0 & 62 & 32 & 154 & 4 & 136 & 60 & 46 \\ 238 & 88 & 46 & 114 & 112 & 62 & 0 & 90 & 284 & 110 & 330 & 232 & 120 \\ 168 & 80 & 68 & 76 & 84 & 32 & 90 & 0 & 250 & 74 & 244 & 160 & 74 \\ 484 & 232 & 172 & 244 & 298 & 154 & 284 & 250 & 0 & 228 & 702 & 482 & 232 \\ 150 & 90 & 80 & 94 & 104 & 4 & 110 & 74 & 228 & 0 & 244 & 164 & 66 \\ 542 & 252 & 222 & 276 & 326 & 136 & 330 & 244 & 702 & 244 & 0 & 540 & 276 \\ 392 & 182 & 150 & 162 & 214 & 60 & 232 & 160 & 482 & 164 & 540 & 0 & 172 \\ 190 & 66 & 52 & 90 & 110 & 46 & 120 & 74 & 232 & 66 & 276 & 172 & 0 \end{pmatrix}, T^b = \begin{pmatrix} 1514 \\ 691 \\ 561 \\ 723 \\ 876 \\ 406 \\ 913 \\ 700 \\ 1881 \\ 704 \\ 2045 \\ 1455 \\ 747 \end{pmatrix}$$

To realize weight annealing within our memristive crossbars, we encode the normalized weights ($T_{ij} < 0$ and $T_j^b > 0$) to device conductances by using $g_{ij} = G_{\max}(T_{ij}/|W^{\max}|)$ and $g_j^b = G_{\max}(T_j^b/|W^{\max}|)$ where $|W^{\max}| = \max\{|T|,|T^b|\}$, where $G_{\max}$ is a predetermined

value. Memristive crossbars inherently implement the dot-product using Ohm and Kirchhoff's laws, i.e.,

$$V(t+1) = \mathrm{f}\left(\sum_{i=1}^{N} g_{ij} V_i(t) + g_j^b V^{\mathrm{ap}}\right). \tag{3.3.2}$$

Note that f(.) is the binary activation function realized by a comparator in peripheral circuits. In our experimental setup, the straightforward peripheral functions are emulated by the Agilent characterization tools. In exponential weight annealing, $V_i(t) = V_i^{\mathrm{ap}}(t)(1 - \mathrm{e}^{\frac{-t}{\tau}})$ is the applied voltage to the i^{th} input and $V_i^{\mathrm{ap}}(t)$ is either 0 or V_{ap}, based on the neuron state. Given that, we rewrite Eq. 3.3.2 as

$$\begin{aligned} V(t+1) &= \mathrm{f}\left(\frac{V^{\mathrm{ap}} G_{\max}}{|W^{\max}|} \sum_{i=1}^{N} T_{ij}(V_i(t)/V^{\mathrm{ap}}) + T_j^{\mathrm{b}}\right) \\ &= \mathrm{f}\left(\frac{V^{\mathrm{ap}} G_{\max}}{|W^{\max}|} \sum_{i=1}^{N} T_{ij}(V_i^{\mathrm{ap}}(t)(1 - e^{\frac{-t}{\tau}})/V^{\mathrm{ap}}) + T_j^b\right) \\ &= \mathrm{f}\left(\frac{V^{\mathrm{ap}} G_{\max}}{|W^{\max}|} \sum_{i=1}^{N} w_{ij}(t)(V_i^{ap}(t)/V^{\mathrm{ap}}) + T_j^b\right), \end{aligned} \tag{3.3.3}$$

which is essentially the hardware implementation of updating the neuron states of the Hopfield model.

After determining the desired conductance map, we program the devices individually using the write-verify algorithm [66]. Note that the details of forming, tuning, and operation of the circuit, as well as the procedure of mapping the actual synaptic weights (from software) to conductance values, are illustrated in the Methods section. Fig. 59b and Fig. 59c show the network's desired weight map and the corresponding conductance map obtained after tuning the crossbar, respectively. Most devices are tuned very close (within 5%) to the desired states,

which is possible due to the tight distribution of switching thresholds in our analog-grade crossbar circuits. Fig. 59d shows the distribution of pre-activation readout currents for the baseline case (the inset indicates no bias in neuron selection). The input "on" voltage corresponding to binary input '1' is V_{ap}=0.1 V. Note that we exponentially increase the "on" applied voltage from 0 to 0.1 V for the weight annealing. The measured synaptic strength of each device during the weight annealing is shown in Fig. 59e. The experimental and simulation results are compared in Fig. 59f. Specifically, the average energy over 10^3 cases for 200 epochs is shown for various methods. Here, the annealing schedule parameters are 10^4, 10^5, and 35 for chaotic, stochastic, and weight annealing, respectively. The ground state locates at -3796, and weight annealing (on both experiment and simulation) performs better than other techniques and far better than the baseline.

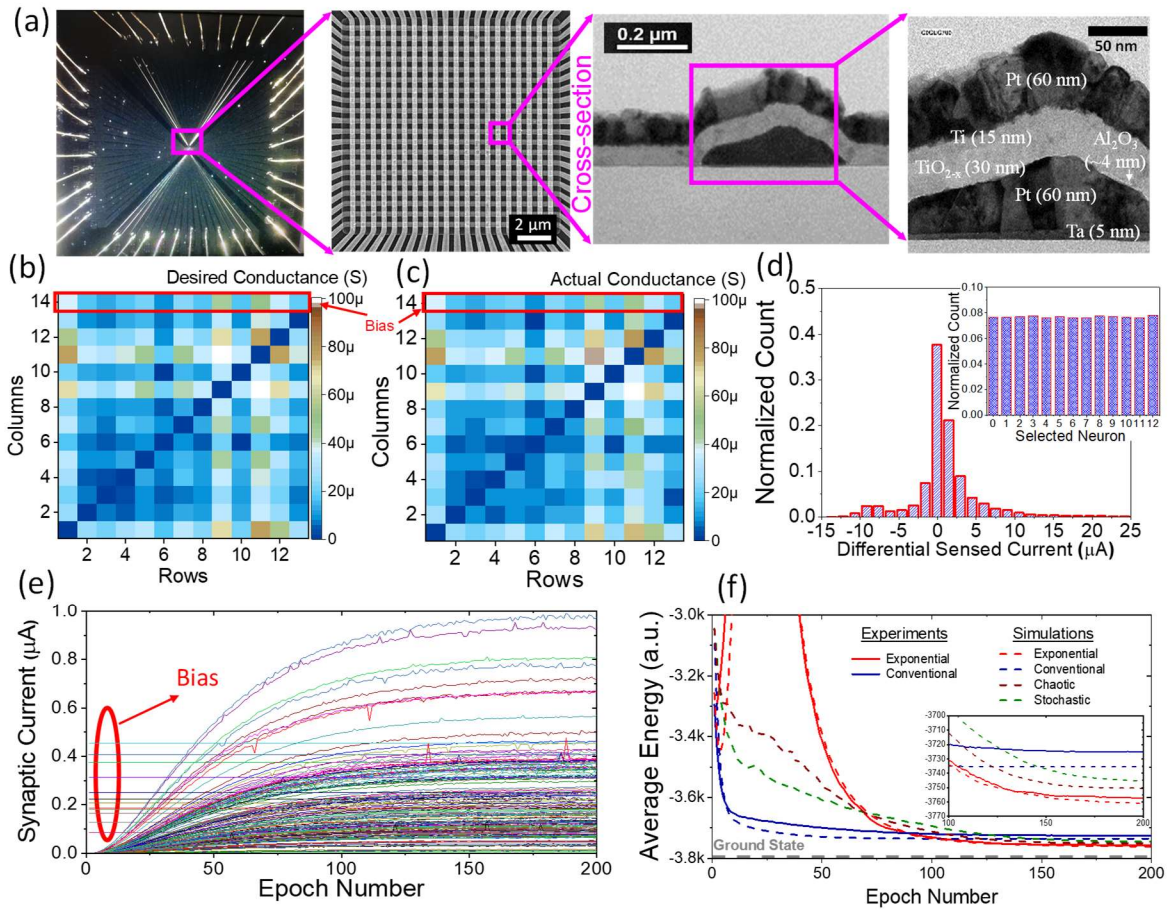


Fig. 59: The experimental demonstration with the integrated memristor crossbars. (a) The fabricated 20×20 integrated memristor crossbar [46]. (b) The desired ideal analog map for the 13-node graph partitioning problem, and (c) the resultant conductance map of the devices after tuning the crossbar. (d) Distribution of the readout current when solving the problem with the conventional (baseline) approach. The inset shows the histogram of selected neurons (for updates) and indicates there is no bias in the neuron update. (e) The evolution of the synaptic weights during the weight annealing. (f) The experimental versus simulation results of the neuro-optimization with different techniques. The inset shows the zoomed-in average energy in the last 100 epochs.

In our second experimental demo, a 7-node maximum-weighted independent set is solved using an array of 12×10 redesigned embedded Flash memories fabricated in GlobalFoundries 55 nm LPe CMOS process (Fig. 60a). The redesigned array structure enables <1% analog programmability [40]. Biasing conditions (imposed during programming) ensure the subthreshold operation of the devices at all operating conditions. Fig. 60b shows the implemented weighted graph. Similar to our first demo, we choose the weights and edges (of

the graph) randomly while ensuring $T_{ij} < 0$ and $T_j^{\mathrm{b}} > 0$. Then, we use the standard method of finding the weights of the Hopfield network by comparing the general Lyapunov energy with the energy function of the problem (see Supplementary Section 2 of Ref. [184] for more information). A graph with the following adjacency matrix and nodal weights are considered:

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}, E = \begin{pmatrix} 6.40 \\ 7.38 \\ 5.05 \\ 1.21 \\ 3.43 \\ 2.02 \\ 6.09 \end{pmatrix}$$

Accordingly, the synaptic weights are found:

$$T = -\begin{pmatrix} 0 & 2 & 2 & 0 & 2 & 2 & 2 \\ 2 & 0 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 0 & 2 & 2 & 0 & 0 \\ 0 & 2 & 2 & 0 & 2 & 2 & 2 \\ 2 & 2 & 2 & 0 & 0 & 2 & 2 \\ 2 & 2 & 0 & 2 & 2 & 0 & 2 \\ 2 & 2 & 0 & 2 & 2 & 2 & 0 \end{pmatrix}, T^b = \begin{pmatrix} 3.20 \\ 3.69 \\ 2.52 \\ 0.60 \\ 1.71 \\ 1.01 \\ 3.04 \end{pmatrix}$$

We can tune each floating-gate cell to a desired current value at nominal biasing conditions. Neglecting the drain-source voltage, $I_{\mathrm{DS}} \approx I_0 e^{\frac{V_{\mathrm{WL}} - V_{\mathrm{th}}}{nV_{\mathrm{th}}}} \approx \theta_0 I_0 e^{\frac{V_{\mathrm{WL}}}{nV_{\mathrm{th}}}}$ where θ_0 is the effective weight of the devices, and other parameters have their usual meaning. We define $I_{ij}(V_{\mathrm{WL}}) = \theta_{ij} I_0 e^{\frac{V_{WL}}{nV_{\mathrm{th}}}}$ and $I_j^b = \theta_j^b I_0 e^{\frac{V_{\mathrm{ap}}}{nV_{\mathrm{th}}}}$. During tunning, we make sure that $I_{ij}(V^{\mathrm{ap}}) = \frac{I_{\max} T_{ij}}{|W^{\max}|}$ and $I_j^b = \frac{I_{\max} T_j^b}{|W^{\max}|}$ where $|W^{\max}| = \max\{|T|, |T^b|\}$. $I_{\max}$ is a predetermined value. Hence,

$$\theta_{ij} = \frac{I_{\max} T_{ij}}{|W^{\max}| I_0} e^{\frac{-V^{\mathrm{ap}}}{nV_{\mathrm{th}}}}, \qquad \theta_j^b = \frac{I_{\max} T_j^b}{|W^{\max}| I_0} e^{\frac{-V^{\mathrm{ap}}}{nV_{\mathrm{th}}}} \tag{3.3.3}$$

The update rule for the j^{th} neuron is given by

$$V(t+1) = \mathrm{f}\left(\sum_{i=1}^{N} I_{ij}(V_i(t)) + I_j^b \right) \tag{3.3.4}$$

and $V_i(t)$ is the gate-applied voltage to the i^{th} input terminal at iteration t. It is effortless to show that 3.3.4 performs the dot-product of (two-state) input voltage vector and (analog) weights. Note that we apply the digital inputs ($V_{\mathrm{WL}} = 0$ V corresponds to ~0 synaptic current

and $V_{\mathrm{WL}} = V^{\mathrm{ap}} = 1.5\ V$ corresponds to the 'on' current for the baseline operation) in the gate-line direction and read the summed currents from the bitlines (e.g., using a transimpedance amplifier or a current conveyor) to $V_{BL} = 1\ V$.

For the exponential weight annealing, we substitute I_{ij} and I_j^{b} in S6 using S5 and obtain

$$V(t+1) = \mathrm{f}\left(\sum_{i=1}^{N} \frac{I_{\max} T_{ij}}{|W^{\max}| I_0} e^{\frac{-V^{\mathrm{ap}}}{nV_{\mathrm{th}}}} I_0 e^{\frac{V_i(t)}{nV_{\mathrm{th}}}} + \frac{I_{\max} T_j^{\mathrm{b}}}{|W^{\max}| I_0} e^{\frac{-V^{\mathrm{ap}}}{nV_{\mathrm{th}}}} I_0 e^{\frac{V_{\mathrm{ap}}}{nV_{\mathrm{th}}}}\right) =$$

$$= \mathrm{f}\left(\frac{I_{\max}}{|W^{\max}|} \sum_{i=1}^{N} T_{ij} e^{\frac{V_i(t) - V^{\mathrm{ap}}}{nV_{\mathrm{th}}}} + T_j^{\mathrm{b}}\right)$$

and $V_i(t) = \frac{V_i^{\mathrm{ap}}(t)t}{\tau}$ (τ is the annealing schedule) for the semi-exponential learning (it is "semi" since the devices have nonlinearities in their *I-V*) and $V_i(t) = V_i^{\mathrm{ap}}(t)(1 - e^{\frac{-t}{\tau}})$ for super-exponential learning. Also, since the currents are extremely small for $V_{WL} < 0.7$, the ground state of the system remains constant for a significant period. This can be prevented by starting from $V_{\mathrm{WL},0} = 0.7$ and, e.g., governing the annealing by $V_i(t) = \frac{(V_i^{ap}(t) - V_{WL,0})t}{\tau} + V_{WL,0}$.

For the considered problem, the ground state of the energy function locates at -5.5755 that corresponds to the neural state "0010001". The devices are programmed with <1% accuracy (see the method section). Fig. 60c shows the resultant map of state currents under nominal biasing conditions, i.e., ($V_{\mathrm{WL}} = 1.5$ V, $V_{\mathrm{CG}} = 2.5$ V, $V_{\mathrm{BL}} = 1$ V, $V_{\mathrm{SL}} = 0$ V, and $V_{\mathrm{EG}} = 0$ V). The experiments and simulations are performed over 128 initialization cases for 500 epochs and show the results in Fig. 60d. The results are averaged over 100 runs in the simulations. The annealing schedule is 10, 10, and 100, and the average probability of hitting the global optimum is 0.76, 0.92, 0.82, and 0.99 for stochastic, chaotic, and weight annealing, respectively (Fig. 60e). We drive the devices corresponding to bias weights (T_j^{b}) by constant gate-voltages ($V_{\mathrm{WL}} = 1.5$ V and $V_{\mathrm{CG}} = 2.5$ V), while other rows (if their corresponding neuron is in the 'on' state) are driven by $V_i(t)$. We studied the impact of annealing schedule and exponential versus linear voltage scaling too and found that a slower annealing schedule

($\tau_{\text{exp}} = 60$) tackles the nonlinearities in the super-exponential dependency of synaptic current to voltage and closely matches the trends in the simulations. For the latter case, the slowest annealing process ($\tau_{\text{exp}} = N_{\text{EP}} = 500$) leads to the best response.

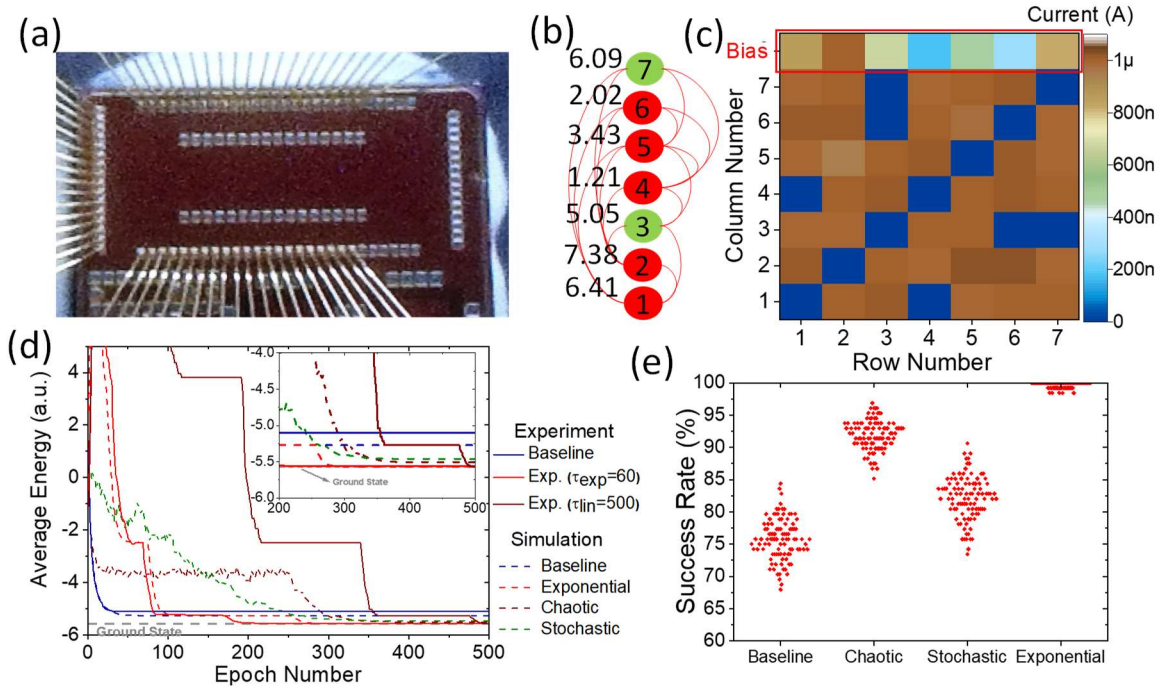


Fig. 60: Neuro-optimization with embedded analog-grade eFlash memories. Panel (a) shows the fabricated 10×12 eFlash array chip in Global Foundries' standard LPe CMOS process [67]. (b) A 7-node maximum-weighted independent set problem. (c) The heat map of the synaptic weights for the devices that implement the neuron-optimization. (d) The average energy versus epoch comparing experimental results with simulations over 100 runs. (e) The success rate of different annealing techniques on this problem over 100 runs.

3.3.3. Methods

In order to increase our demo size (given our 20×20 crossbar size), we deliberately chose edges to be larger than weights (the values are selected randomly in all experiments and simulations) to force all non-diagonal synaptic weights (T_{ij}) to be negative and all biases to be positive. This technique allows us to implement a relatively larger demo by assigning one device per weight (compared with the two-device per synapse needed for fully differential design) and perform each vector-by-vector multiplication in two cycles. Indeed, we

implement the dot-product operation in a two-step time-multiplexed fashion: In one cycle, we measure the total current ($\sum I^-$) associated with the input vector multiplied by the synaptic weight vector (from the selected neuron), while the input bias voltage is zero. Then, we subtract it from the sensed current ($\sum I^+$) from the same bitline, while the main inputs are zero and apply V_{ap}=0.1 V to the bias column. Besides, to increase the dynamic range, all bias conductances are divided by 5 and compensated by applying an extra gain of 5 at the neuron side. In other words, we evaluate the final output by hard thresholding ($5\sum I^+ - \sum I^-$). (Note that we have previously demonstrated fully-differential dot-product engines using the same devices in our previous works – see, e.g., [9]).

Owing to the single-ended design, we use $g_{ij} = G_{\max}(T_{ij}/\max(|T_{\max}|))$, where $\max(|T_{\max}|)$ is the maximum absolute weight and $G_{\max}$ is the maximum absolute conductance (40 μS in our experiment). We ground all bitlines (bottom electrode) except the one associated with the selected neuron. That is virtually grounded, and we sense and sample its current at 1 MHz using a B1530A fast measurement unit and a B1500A parameter analyzer. We apply neuron voltages to the switch matrix, connected to both 20 rows and 20 columns of the crossbar. We link top electrodes to the input neurons and bottom electrodes to the output neurons through an E5250A switch matrix.

The eFlash chip, fabricated in GlobalFoundries 55 nm LPe process, includes a 12×10 redesigned industry-grade split-gate memory array. The packaged chip is previously used for developing a high-performance dot-product engine. Agilent B1500A and B1530A tools are used for measurements and pulse generation. We have developed a custom-made switch matrix on a printed circuit board controlled via a lightweight microprocessor to interface Agilent tools with the chip. More details on the experimental setup, programming, eraser,

redesigned layout structure, half-select disturbance immunity, retention, and endurance characteristics are available in Ref. [40]. All eFlash memories are programmed to their targeted states at $V_{\mathrm{WL}} = 1.5\ V, V_{\mathrm{CG}} = 2.5\ V, V_{\mathrm{BL}} = 1\ V, V_{\mathrm{SL}} = 0\ V,$ and $V_{\mathrm{EG}} = 0\ V$ and operated at the same biasing condition. Further, the devices are tuned one at a time by progressively increasing voltage pulses and using the write-verify algorithm. We have discussed the details of pulse amplitudes and durations in the programming phase in Ref. [40].

The weight annealing is implemented by increasing V_{WL} from 0 to 1.5 V linearly and exponentially. This would exponentially and superexponentially increase the synaptic weights since we operate the devices in weak inversion, as discussed before. Similar to the other demo, we use the single device per synapse topology and compute each update in two cycles. The weights are mapped (from software to hardware) by using $I_{ij}^{T} = I_{\max}\frac{T_{ij}}{|T_{\max}|}$ and $I_{j}^{\mathrm{b}} = I_{\max}\frac{T_{j}^{b}}{|T_{max}^{b}|}$ in which $I_{\max} = 1\ \mu A$, $T_{\max} = 2$ is the maximum absolute synaptic weight, and $T_{max}^{b} = 3.694$ is the maximum absolute bias.

3.3.4. Discussion and summary

We have demonstrated weight annealing, a technique that substantially improves the speed and accuracy of asynchronous Hopfield neuro-optimizer. The weight annealing converges faster and to a better solution within the studied runtime as compared to other considered annealing approaches. We highlighted the scalability of our approach with the problem size and computational time by exploring several combinatorial problems and demonstrated the experimental results using two state-of-the-art analog-grade nonvolatile memories.

The passive integrated memristor technology offers the best scaling prospects and low fabrication cost. We have recently developed a 4K fully CMOS-compatible 0T1R array with

excellent switching characteristics [113]. The measured analog characteristics are promising for the development of large-scale neuro-optimization systems. On the other hand, eFlash technology is much sparser, but it is currently commercially available and embedded in (down to 28 nm) standard CMOS processes. Our preliminary estimations indicate a promising prospect for using metal-oxide memristors for the hardware implementation of Hopfield networks and weight annealing. Our approach could be $10^2\times$ faster and $10^5\times$ more energy efficient than the most efficient conventional methods based on graphics processor units on the same task. Future works focus on the CMOS-integrated design of a weight annealing optimizer, allowing us to perform a rigorous comparison with entirely fabricated annealing machines.

As opposed to most previous works that focus on large-scale switching of memristors, our proposed solution offers very infrequent writes, which is justified assuming long runtimes of computationally-extensive problems. More importantly, our proposed neuro-optimizer offers analog (>5 bits with memristive nanodevices and >6 bits via eFlash technology) weights. This feature is not demonstrated in previous Ising machines [46,47,48,186-189]. Unlike quantum computing machines [45] that are susceptible to environmental noise, hard to scale, and must operate at cryogenic temperatures, the proposed circuit is more scalable and operates at room temperatures.

2.13. Mixed-Signal Neuro-Optimization with Adaptable Annealing

In section 2, we demonstrated the idea of using intrinsic and extrinsic analog noise for building a stochastic mixed-signal dot-product engine. We also illustrated how chaotic annealing could be realized using the main-stream analog-grade nonvolatile memories. In chapter 3, we proposed the idea of weight annealing, which shows excellent performance and

could be easily implemented via crossbars of memristive devices. In addition, in both previous sections, only single combinatorial problems were implemented in hardware, and a small-scale memristive crossbar was used for the demonstration. Here, we demonstrate analog neuro-optimization hardware, suitable for solving a large number of combinatorial optimization problems, based on a crossbar circuit with 4096 passively-integrated analog-grade memristors, which was thoroughly elaborated in section 2. The proposed hardware supports a variety of metaheuristic techniques for improving optimization performance, such as stochastic, chaotic, weight annealing. The hardware operation is successfully tested by experimentally solving weighted graph partitioning, maximum clique, vertex cover, and independent set problems and observing good agreement with simulation results.

3.4.1. Neuro-Optimization Hardware

Fig. 61a shows the main idea of the proposed mixed-signal discrete-time/state stochastic Hopfield neural network circuits, essentially mixing the described techniques in sections 3.2 ,3.3. to support simulated, chaotic, and weight annealing. This circuit is verified with the 64×64 crossbar circuit, which is packaged, integrated with the experimental setup, and used for all the measurements (Fig. 61b).

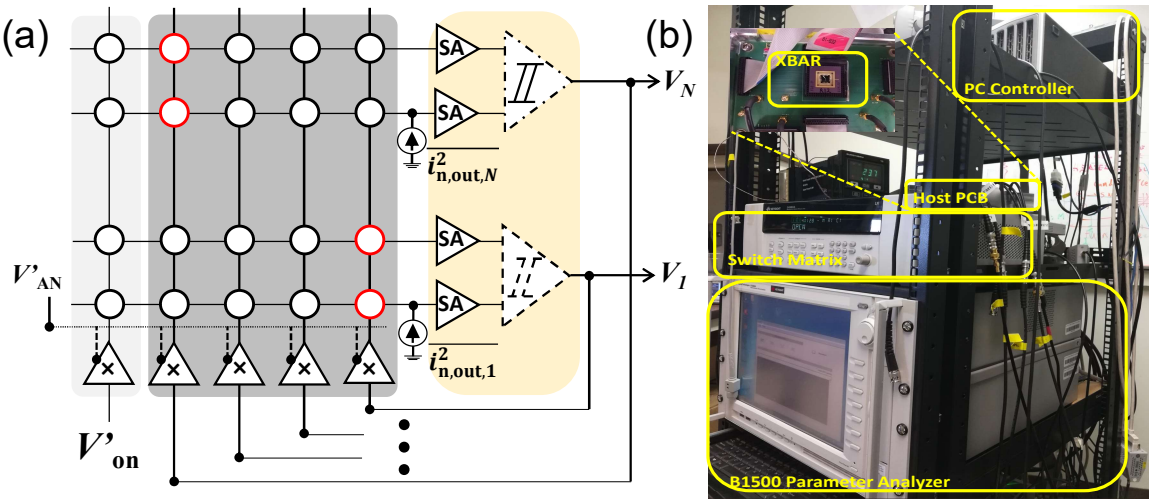


Fig. 61: (a) Circuit diagram of the versatile neuro-optimizer. Two sense amplifiers and a comparator represent a neuron. (b) The measurement setup of the RRAM-based neuro-optimizer.

The stochastic dot-product computation with adjustable annealing schedule is implemented by controlling the signal-to-noise ratio of the read-out current. Specifically, the output-referred current noise on each differential line ($\overline{\iota_{n,out}^2}$) is the sum of current noises generated by the memory cells and that of peripheral circuits. For the practical operational frequencies (>100 MHz), such noise is predominantly thermal. The comparator implements the step function on a sampled current so that the probability of latching a digital '1' value is $0.5(1 + \mathrm{erf}\ \langle I\rangle/\sqrt{2}\sigma)$, where σ is the standard deviation of the output current. The error function matches closely, with relative error always within 2% across the whole range of normalized currents, probabilistic neuron transfer function 1/(1+exp[-y/T]) commonly used in stochastic Hopfield neural networks, in which y is the pre-activation value and T is the temperature. Because $\overline{\iota_{n,out}^2}$ is independent of the applied voltages and is contributed by all differential line memristors and the sensing circuit, the effective temperature is inversely proportional to the peak signal-to-noise ratio, i.e., $\mathrm{SNR_{max}} = I_{\mathrm{max}}/\sigma$. In turn, such design enables very compact SSA implementation (Fig. 61a), in which the temperature can be controlled by altering V_{ON} (and hence modulating I_{max} and $\mathrm{SNR_{max}}$), which is the amplitude of the applied 'on' voltage to the inputs of the crossbar circuit during the operation (Fig. 61a).

To implement chaotic annealing, we initially set the neuron self-feedback weights to some large values (as compared to other weights in the network), which results in chaotic dynamics of the network. These weights are then exponentially decreased to zero during the operation so that the network slowly transitions from chaotic to the periodic regime, eventually settling in a stable equilibrium. The weight adjustment is performed again by scaling the applied

voltages. Because the voltages cannot be scaled for only diagonal devices without affecting other devices on the same columns, the chaotic approach requires doubling the number of columns and setting all but diagonal weights to zero in the additional crossbar array. The neuron switching activity (firing rate) for simulated, chaotic, and baseline (i.e., with no annealing) approaches was studied using a 10-node weighted graph partitioning problem with randomly initialized weights. The experimental results show that the neuron firing rates were initially above 5% for both simulated and chaotic, corroborating the random switching of neurons but eventually reduced to zero when the network stabilized.

Furthermore, for the weight annealing, the weights are defined as $W_{ij}(t)= T_{ij}(1-\exp[-t/\tau])$, where t is epoch (i.e., one neuron update) number, τ is the annealing factor, and T_{ij} is the predetermined weight matrix corresponding to the problem in question. The network weights are then slowly modified to the baseline ones, with the goal of always keeping the network in the ground state during the transition. Such behavior of weight annealing approach is experimentally confirmed on 10-node partitioning problem (Fig. 62b) using the same (as in Fig. 62a experiment) weights. Fig. 62c shows the final average energy of the network for all studies techniques on the same problem, in particular showing >2× higher convergence rate to top 5 solutions for weight annealing over naive baseline approach. The experimental results also confirm that the weight scaling error due to I-V nonlinearity (which could lead to varying relative weights when changing V_{ON}) is negligible.

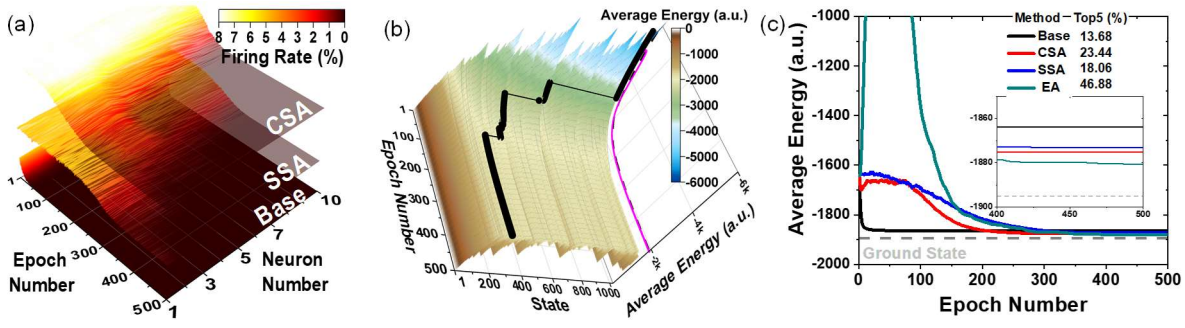


Fig. 62: Mechanism of baseline, simulated stochastic annealing (SSA), and simulated chaotic annealing and weight annealing techniques described by solving a graph partitioning problem. (a) The firing rate of a 10-node graph partitioning problem (with uniformly distributed weights in the range of 1-20) for different techniques. (b) Weight annealing dynamics ($\tau = 100$) in solving the 10-node weighted graph partitioning problem. The network finds the ground state at earlier epochs and tracks it while transitioning to the baseline. Black: the transitory ground state of the system, magenta: average energy over 50k cases.

3.4.2. Solving Combinatorial Optimization Problems

To further demonstrate the effectiveness of the proposed hardware, we solve common combinatorial optimization problems and study a certain parameter in each case.

For the first, we solve 10 random configurations of 5-node weighted maximum-clique problems. The problem parameters are converted to network weights, which are then mapped to devices with <5% tuning error in the range of 10 kΩ to 150 kΩ. The annealing parameter is exponentially scaled from 1 to 0.01 for stochastic and from 20 to 0.1 for chaotic, while τ = 20 for weight annealing. All three annealing techniques performed much better than the baseline, with good agreement between simulations and experiment (Fig. 63). The simulation results for a larger number (200) of random configurations indicate better weight annealing performance.

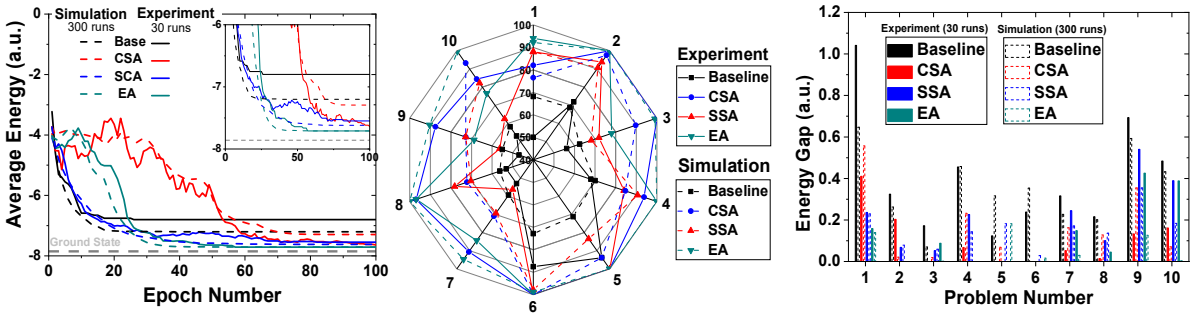


Fig. 63: Solving 5-node weighted maximum-clique problems with memristive crossbars. (a) The measured evolution of the average energy for a 5-node maximum-weighted clique problem. The inset shows the zoomed-in view of the figure. (b) The success rate of different annealing techniques on 10 5-node weighted maximum clique problems. (c) The energy gap (the average energy of solution minus the ground state) for all 5-node weighted maximum clique problems.

Figs. 64 shows the results for the weighted vertex cover problem. The performance of annealing approaches, measured on various size graph problems, is always better compared to the baseline. The impact of the annealing schedule is studied by solving a 10-node independent set problem. The results are presented in Fig. 64, which shows a significant improvement for the average energy when using slower annealing. Finally, we considered solving the graph partitioning optimization problem for a fully-connected weighted 6-node graph. Fig. 65 shows the final experimental results and their comparison to simulations, further validating the proposed hardware functionality and >20% improvements in the success rate over the baseline approach.

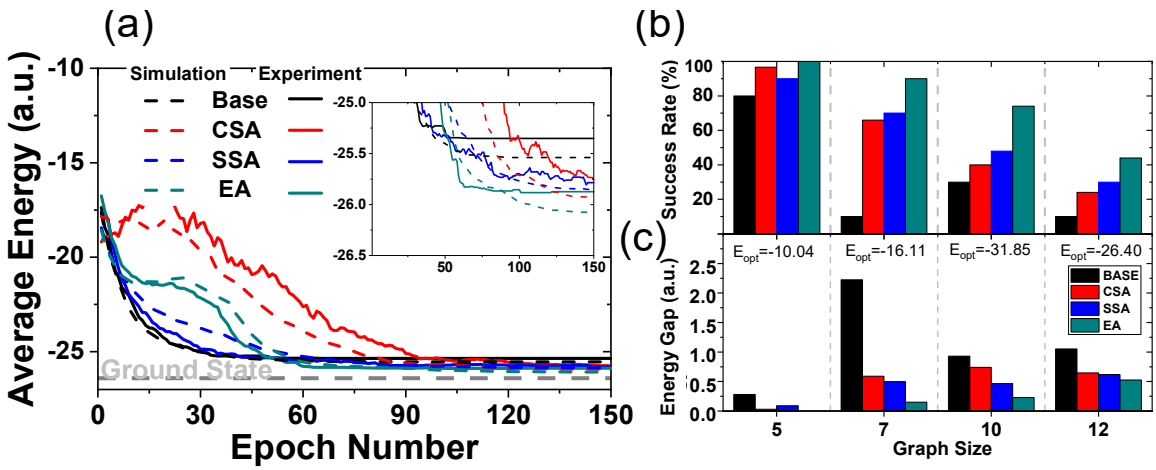


Fig. 64: (a) Experimental (30 runs) versus simulation (300 runs) results of solving a 12-node minimum-weighted vertex cover. Inset is the zoomed-in view. Experimental results on various sizes of minimum-weighted vertex cover problems: (b) the success rate and (c) the energy gap.

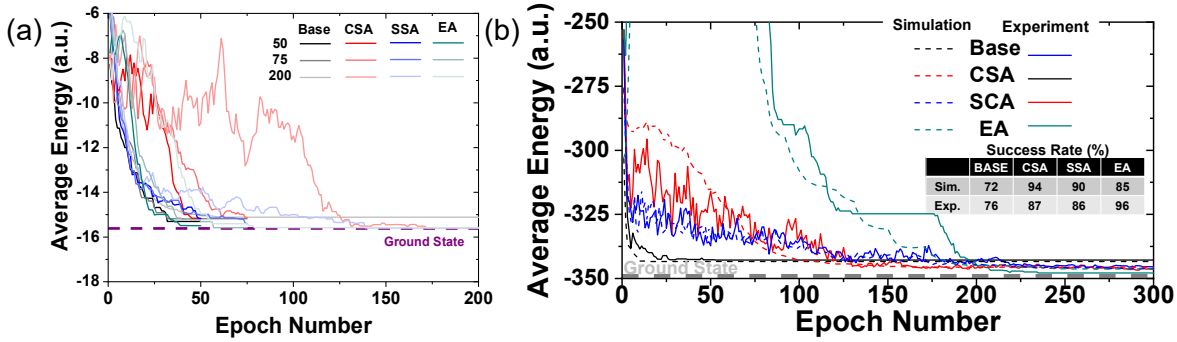


Fig. 65: (a) The measured average energy (over 30 runs) when solving a 10-node maximum-weight independent set problem with three different annealing schedules. (b) The average

energy (300 and 30 cases for simulation and experiment, respectively) of the neuro-optimizer when solving a 10-node graph partitioning problem.

Therefore, this section showed that our memristor-based neuro-optimizer could be used to implement 3 different annealing techniques: simulated annealing, chaotic annealing, and weight annealing. Besides, it was demonstrated that this circuit could be used to efficiently solve various combinatorial optimization problems, including weighted graph partitioning, minimum-weighted vertex cover, maximum-weighed clique, and maximum-weighted independent-set problems. The experimental results showed improved performance for all annealing techniques over the baseline approach and superior performance of exponential annealing compared to stochastic and chaotic annealing.

In previous sections, we focused on designing efficient mixed-signal hardware for high-speed and energy-efficient implementation of Hopfield neural networks and Restricted Boltzmann machines, which find applications in solving combinatorial optimization, dimensionality reduction, etc.

2.14. Summary and Future Works

In the first three sections of this chapter, we discussed novel techniques to design mixed-signal computing circuits targeting neurooptimization and neurocomputing applications. First, we proposed to utilize extrinsic and intrinsic noise sources in mixed-signal memory-based circuits to implement efficient stochastic dot-product operation with runtime adjustable temperature. We then experimentally verified this idea by demonstrating memristive restricted Boltzmann machine and solving a combinatorial optimization problem with floating-gate memory neuromorphic circuits. In the following, we proposed the novel weight annealing technique that boosts the performance of the Hopfield model in solving combinatorial optimization problems. Using extensive simulations on four representative problems, we

numerically demonstrate that the proposed method outperforms the conventional Hopfield network (baseline) and challenges the prominent stochastic and chaotic annealing techniques in both computational time and accuracy. Then, we showed an efficient, scalable, and fast circuit implementation and experimentally verified it in two memory technologies: integrated 20×20 0T1R memristive nanodevices and 12×10 redesigned commercial embedded flash memories. Finally, we demonstrated analog neuro-optimization hardware, suitable for solving a large number of combinatorial optimization problems, based on a crossbar circuit with 4096 passively-integrated analog-grade memristors. The proposed hardware supports a variety of metaheuristic techniques for improving optimization performance, such as stochastic and chaotic simulated annealing and weight annealing. The hardware operation is successfully tested by experimentally solving weighted graph partitioning, maximum clique, vertex cover, and independent set problems and observing good agreement with simulation results.

We believe that future experimental work should focus on more promising continuous-time/state networks with parallel state updates [201] based on fully integrated hardware. The most urgent theoretical work includes modeling the impact of the circuit and device non-idealities on the network functional performance, carrying out a more rigorous comparison of annealing techniques for neurooptimization, and developing larger-scale hardware suitable for more practical applications. In this context, it is worth mentioning that for the hardest combinatorial optimization problems, such as maximum clique problems, finding even a largely suboptimal solution is challenging, which could significantly relax the device and circuit requirements.

Spiking neural networks, the most realistic artificial representation of biological nervous systems, are also promising due to their inherent local training rules that enable low-overhead

online learning and energy-efficient information encoding. Their downside is the more demanding functionality of the artificial synapses, notably spike-timing-dependent plasticity, making their compact, efficient hardware implementation challenging with conventional device technologies. Nonvolatile memories, and in particular, ultra-scalable memristors, are excellent candidates for artificial synapses because practically valuable neural networks require a massive number of synapses. We have performed a preliminary experimental study of demonstrating coincidence detection using a spiking neural network, implemented with passively integrated metal-oxide memristive synapses connected to an analog leaky-integrate-and-fire silicon neuron [9]. By employing spike-timing-dependent plasticity learning, the network can robustly detect the coincidence by selectively increasing the synaptic efficacies corresponding to the synchronized inputs. Not surprisingly, our results indicate that device-to-device variation is the main challenge in realizing more complex spiking networks.

4. Hardware Security Primitives with Analog Memories

The proliferation of networked mobile devices and smart gadgets in the IoT landscape has accelerated the demand for lightweight, low-power, and operationally compatible cryptographic solutions. As a result, emergent hardware-intrinsic security architecture must demonstrate manufacturability, power efficiency, and platform compatibility, in addition to robust security performance. In this chapter, we show that the analog tuning and nonlinear conductance variations of memristors can be used to build a fundamental building block for implementing physically unclonable functions that are resilient, dense, fast, and energy-efficient. Using two vertically integrated 10×10 metal-oxide memristive crossbar circuits, we experimentally demonstrate security primitive that offers a near-ideal functional performance. Then, we propose RX-PUF that takes advantage of a two-step readout scheme to avoid the need for the conductance tuning procedure used in the original design. Specifically, a 600 kb challenge-response pair (CRP) PUF using 250 nm half-pitch 20×20 crossbar arrays with passively integrated devices. The measured bit error rate is 0.7% at RT and ≤ 5.3% at 100°C, even without using any error correction methods. The measured responses show near-ideal uniformity (50.04%) and inter-HD (50.12%) and pass all relevant National Institute of Standards and Technology (NIST) randomness tests.

Further, we present a novel architecture, called VRPUF, and prototype it using unformed 4K-ReRAM passive crossbar circuits fabricated with a CMOS-compatible process, suitable for the back-end-of-line (BEOL) integration. The architecture utilizes intrinsic process variations in crossbar circuits, manifested as variations in device I-V nonlinearities and the leakage currents, and allows for a huge (~10^{25}) number of challenge-response pairs (CRPs). The VRPUF design does not require forming/ programming crosspoint devices, which

simplifies peripheral circuits, leading to ~ 4× better density compared to the architectures which rely on switching the states of ReRAM devices. Moreover, uniform *IV*s of the virgin-state devices, coupled with lower conductance and stronger static nonlinearity, allow for ~100× improvement in power consumption and more robust security metrics. The intrinsic reliability of VRPUF primitives is also ~4× better compared to CMOS architectures. Moreover, a novel leakage injection approach using an electrically isolated portion of the crossbar array is proposed to boost the PUF's robustness, and a key-booking scheme is introduced, which dramatically improves reliability across a wide temperature range of operation and further increases PUF circuit density by reducing error-correcting overheads.

We then introduce ChipSecure, a PUF architecture based on eFlash memories that exploit randomness in static *I-V* characteristics and reconfigurability of embedded flash memories to design efficient physically unclonable function. Leakage current and subthreshold slope variations, nonlinearity, nondeterministic tuning error, and sneak path current in the redesigned commercial flash memory arrays are exploited to create a unique digital fingerprint. A time-multiplexed architecture is designed to enhance the security and expand the challenge-response pair space to 10^{211}. Experimental results demonstrate 50.3% average uniformity, 49.99% average diffuseness, and native <5% bit error rate. The analysis of the measured data also shows strong resilience against machine learning attacks and energy-efficient operation. Accelerated aging measurements indicate stable physical unclonable function response after 900 minutes of baking at 85°C.

4.1. Introduction

The Internet of Things (IoT) enables ubiquitous sensing, which finds enormous applications in modern life. Indeed, the ability to understand environmental indicators in a

network of communicating-actuating devices has revolutionized technology and life. In such an unprecedented proliferation of connected devices, platforms, sensors, and *things*, information is endlessly carried over shared and remotely accessible networks. Hence, provisioning security services are among the most major concerns. Additionally, IoT technology is driven by extremely low-power and low-cost devices, which we rely on for handling sensitive information [202]. Securing an energy-harvested integrated system in a limited area and budget is an ongoing problem and challenging optimization task.

Identification and authentication are the common security tasks in IoT devices implemented with cryptographic primitives. The former is realized in the form of stored keys in a nonvolatile memory (NVM) or electronic IDs, while the latter is implemented with a rather more sophisticated one-way or mutual protocol in which a verifier entity asks a predetermined question and the prover responds accordingly [203]. Block ciphers, true random number generators (TRNGs), and secret key storage are perhaps the most widely used security primitives in IoT devices, and their power-efficient, compact, and secure implementation is a quest, which is actively pursued. For example, privacy-preserving mutual authentication is a specific protocol based on advanced encryption standard algorithm in which PUF can be used for the secret key generator and storage, while TRNG is utilized to generate nonce [204].

While block ciphers have been unsurprisingly designed almost exclusively in digital CMOS technologies, recently, several promising secret key generators, TRNGs, and lightweight authentication solutions have been demonstrated based on NVMs, including emerging nano-scale memories. These devices offer various advantages over conventional CMOS not just due to their superior scalability prospects but because of reconfigurability and

low-cost local computing capability. Besides, stochastic switching in NVMs is a rich entropy source. While such properties have been extensively studied and applied in the context of computing applications, recently, they have been leveraged in building novel promising security primitive circuits as well. In a typical scenario, reconfigurability could be useful when the end-user needs a new key. This is when the original one has been revealed, or the ownership is revoked or updated. Analog-grade NVMs offer potentially better security prospects considering local computing capabilities such as nonlinear characteristics and multi-bit storage capacities. On the other hand, retaining the stored information during power shut down, NVMs may pose additional challenges to data protection [74]. With all pros and cons, flexibilities, and vulnerabilities, mature or emerging NVM technologies bring new options, features, and alternatives to the table, which are undoubtedly worth investigating.

Though the idea of using random physical features for identification is not new, the concept of PUF has been formalized only recently as a (reproducible, unique, evaluable, unclonable, one-way, and tamper-evident [205]) crypto primitive which leverage the intrinsic (nanoscopic physical) variations of a system to generate unclonable secrets. Thus, PUF circuits generate a unique response even if they are similar by design and layout. On the other hand, unlike TRNG, a specific PUF instance is required to generate a similar response to the repeated application of the same challenge. A PUF instance stimulated with a challenge C_s generates a corresponding response R_s. The tuples (R_s,C_s) are often both binary numbers and called a challenge-response pair (CRP). In NVM-based PUFs, for instance, a challenge is either applied to decoders for selection of a certain device(s) or to switching circuits for biasing some rows/columns. The response is obtained, for example, by digitizing the analog sensed current or by comparing it with a certain reference current. The capacity of a PUF is

determined by the total number of CRPs, which is often (either linearly or exponentially) a function of memory array size. PUF security is a measure of resilience against modeling attacks (e.g., using machine learning tools), which could lead to practical ways of reproducing PUF functionality. The indirect (necessary but not sufficient) measures of security are uniqueness, which is a measure of the dissimilarities among keys generated by different PUFs, and diffuseness, which is a measure of the dissimilarities among keys generated by a certain PUF, uniformity which is an indication of balanced 1s and 0s in keys, and reliability which is a manifestation of stable key generation under extreme operating conditions.

To achieve this goal, for M instances of PUF and based on a set of randomly selected challenges, 64-bit or 128-bit keys K_i^j where $1 \leq i \leq N$ (the number of keys) and $1 \leq j \leq M$ are typically evaluated *T* times under varying supply voltage, temperature, and extreme baking conditions. Then, uniqueness is determined by finding the distribution of fractional Hamming distance (FHD), the normalized number of dissimilar bits between two keys, among different keys generated by different PUF instances, i.e., the distribution of $FHD_i = FHD(K_i^j, K_i^p)$. Diffuseness is calculated by finding the distribution of FHD among *N* keys generated by a specific PUF device, i.e., the distribution of $FHD_j = FHD(K_i^j, K_l^j)$. Uniformity is obtained by finding the distribution of fractional Hamming weight (FHW), the normalized number of 1s in a key among *N* keys generated by a specific instance. Finally, reliability is assessed in terms of BER, which is essentially the relative number of bit flips (or the normalized Hamming distance) among *N* keys, generated by applying the same challenge to a specific PUF instance (*T* times) under different operating conditions. In an ideal PUF, uniqueness, diffuseness, and uniformity are 50%, and a reliable PUF has zero bit-error rate. The average and standard deviation of such pseudo-Gaussian distributions are typically used to compare

the quality of PUFs. The other important criteria, particularly for PUFs applied in the authentication of edge devices, are energy and area efficiencies. The former is defined as the energy dissipated for generating a single bit response. The latter is defined as the active area, in terms of F^2 (where F is the minimum feature size of the process technology), consumed for generating a single response bit. Throughput, which is defined as the bit generation rate, is also an important parameter, particularly in key generation applications.

PUFs are typically classified as weak and strong based on security performance. The former is used to generate and store secret keys and feature a relatively small access-restricted CRP space. For a weak PUF, a complete mapping function can be deduced by observing a limited number of CRPs. Strong PUFs, on the other hand, are used for authentication applications, and a complex mapping behavior is constructed by incorporating many nonlinear random components. Strong PUFs have a large CRP space and are unpredictable and resilient toward modeling attacks. In the simplest low-cost protocol, PUFs are utilized in two phases: namely, the enrollment in which a certain number of CRPs are evaluated and stored in a secure database and verification in which PUFs are deployed and used in the field. The NVM-based PUFs, however, often need a preprocessing step to prepare or configure the instance before enrollment (Fig. 66). For example, most emerging memory devices need to be formed before programming. Regardless, NVM-based PUFs are often operated in a pre-configuration phase to optimize certain evaluation metrics (e.g., reliability). For key generator PUFs, the optimization is typically guided towards minimizing the BER. For PUFs employed in authentication, the focus is typically on the security and unpredictability of generated keys.

For PUFs, unclonability is perhaps the most challenging assessment and requirement to meet. Indeed, designing a primitive block that is both physically and mathematically

unclonable is challenging yet crucial for authentication applications. In such cases, the BER requirement is relaxed since the PUF output is directly used to authenticate the prover (and within a certain margin, the identification (ID) can be verified). The CRP space should be large enough to allow the parties to change the challenge after each run and prevent the possibility of man-in-the-middle attacks. On the other hand, there should be enough entropy in the IDs to allow the unique identification of each party.

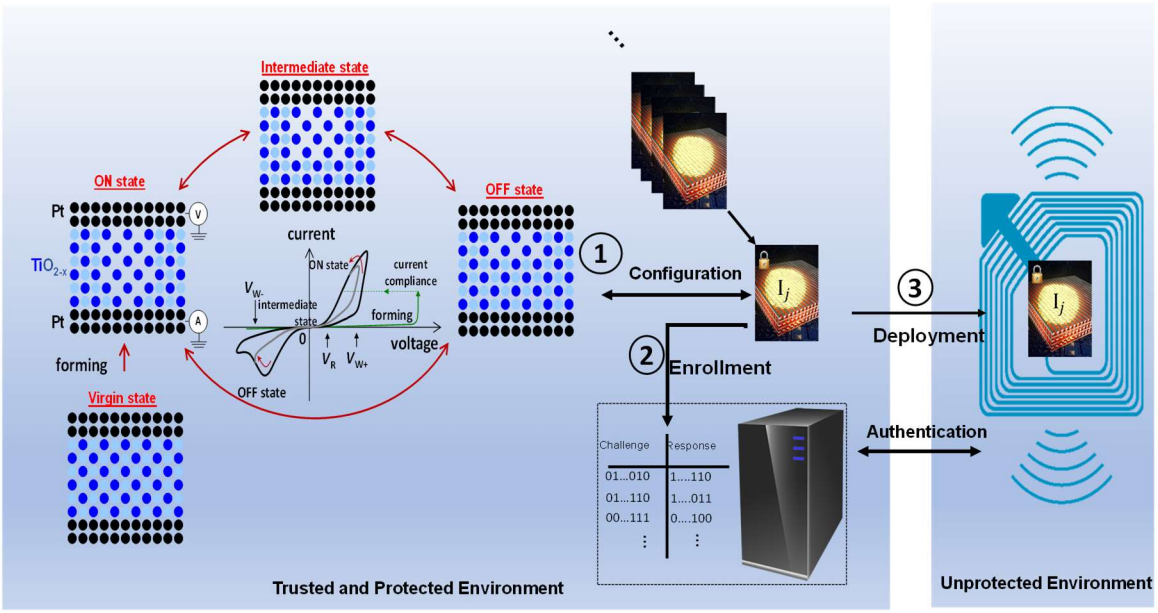


Fig. 66: The simple three-step application scenario for using NVM-based PUF: In the production line, the memristor is formed and configured before enrollment and deployment.

The term *strong PUF* is consequently associated with the designs which typically possess a considerable capacity, often an exponential function of the block size. However, more importantly, the condition, which is often not straightforward to prove, is mathematical unclonability. Naively speaking, it means resiliency toward modeling attacks given the adversary has access to the device and can eavesdrop/probe a certain number of CRPs. Also, PUF's security cannot be ensured by obfuscation of the design because an adversary may reverse-engineer all the circuit details. We also emphasize that in most applications, the demonstration of a significant number of unpredictable CRPs through computational models

suffices since true randomness may not be easy to prove. For example, machine learning models, trained and applied properly, are powerful tools that have been proven to be effective in recent years and are becoming mainstream tools for attacking security primitives. Interestingly, most proposed CMOS strong PUFs have been easily compromised by machine learning attacks. For example, Arbiter PUF and its variant were successfully attacked with a Support Vector Machine (SVM) and a multilayer perceptron [203,206-208]. Power Grid PUFs are also examples of CMOS strong PUFs, which can be compromised too, due to its simple linear model similar to many other designs [209]. It is noteworthy that Shannon's security paradigms - diffusion and confusion, state that highly nonlinear computational elements should be used, and each output must have precisely the same correlation with all the inputs to have a complex system. Interestingly, confusion and diffusion are rarely applied in the creation of CMOS PUFs [210]. NVMs provide us with the opportunity to tackle the modeling problems guided by Shannon's security paradigms.

The most accessible manifestation of process-induced compositional and structural variations in memristor arrays is the spatial (that is, device-to-device) variations of the effective switching thresholds. One example is the voltage at which device conductance is abruptly changed upon application of a ramping bias. A related example is spatial variations in the ON and OFF state conductances in the array upon application of a large voltage or current bias. The physical source of these variations is arguably the stochastic nature of ionic switching arising from compositional inhomogeneity of the switching medium, as well as variations in individual device profiles such as electrode imperfections and random variations in surface roughness. These "entropy" sources are typically the foundation for previously proposed memristor-based security primitives (see [73] for a comprehensive review). Many

of these proposed PUFs require a relatively large number of devices in the crossbar array and extensive peripheral programming and control circuitry to achieve viable operational metrics. Furthermore, a digital mode of operation with devices switched to the extreme ON and OFF states is typically utilized, hence ignoring one of the main advantages of memristive devices: their nonlinear adjustable *I*-*V*s. Indeed, because the device nonlinearity is strongly dependent on the memory state and is correlated with process variations, it can serve as a prominent source of entropy in memristive arrays.

The basic building block for our security hardware is implemented with a two-level stack of monolithically-integrated fully passive Al_2O_3/TiO_{2-x} 10×10 memristive arrays. The fabrication flow ensures a high device yield (>95%) and low <175 °C temperature budget, compatible with CMOS back-end-of-line integration. Fig. 67 shows how such an effective $M \times N = 20 \times 10$ crossbar circuit with crosspoint device conductances G_{ij} is utilized to implement basic cryptographic functionality. A single-bit binary output b is calculated by biasing m selected rows with voltage V_B and then comparing the currents running into two groups of $n/2$ selected virtually grounded columns. For simplicity, let us assume that one group always comprises the leftmost columns and the other the rightmost so that $b = \begin{cases} 1, & I^+ > I^- \\ 0, & I^+ \le I^- \end{cases}$ and $I^{\pm} = V_B \sum_{j \in S_C^{\pm}} \sum_{i \in S_R} G_{ij}(V_B)$ where S_R is a set of indexes of the selected rows, S_C^+ and S_C^- are the sets of indexes of the selected columns in the left and right groups, respectively, and I^+ and I^- are their respective currents. The remaining (unselected) rows and columns in the array are kept floating. With such a scheme, the maximum number of distinct selections is $C_{MAX} = \binom{M}{m} \times \binom{N}{n}$.

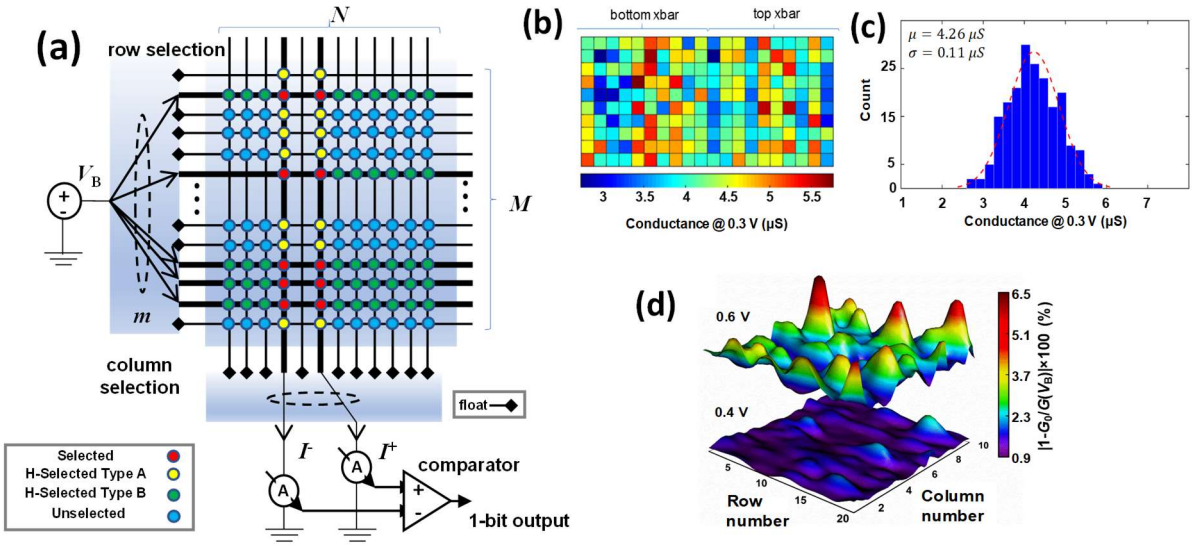


Fig. 67: Memristor-based basic building block for cryptographic hardware [14]: The one-bit output is generated by applying a voltage bias to m rows (of M total) and then comparing the total currents running into the two selected groups comprised of n columns (of N total). In the simplest implementation, the unselected rows and columns in the array are kept floating. (b) Conductance map (G_0), (c) corresponding histogram, and (d) nonlinearity factor for two values of V_B for all 200 devices after tuning. In panel c, the dashed line is a guide showing a Gaussian distribution.

The exemplary PUF circuit is implemented by tuning the conductances of the devices to pre-calculated values using the write-verify algorithm, with the goal of enhancing the contribution of *I-V* variations to the response while improving the reliability and randomness. This is achieved by selecting a specific distribution of device conductances and having a proper balance between two types of currents measured at the output - currents via selected devices and sneak path currents passing through the floating portion of the array. The exact optimization algorithm for finding the conductance distribution and configuring the crossbar circuit is discussed in Ref. [14]. The resultant narrow distribution of device conductances improves the uniformity and diffusiveness by eliminating biases in the output currents. At the same time, the reliability (BER) of the PUF, in particular its tolerance to the memristors' current fluctuations due to intrinsic noise and potential drift of conductive states, is strengthened by enforcing the current readout margins.

Fig. 67b-c shows the results of tuning the memristors' conductance to the values determined by the algorithm. As expected, the target and the tuned conductance distributions were Gaussian-shaped (Figs. 67c). The distribution of relative nonlinearity at 0.4 V and 0.6 V biased is also shown in Fig. 67b-c, indicating larger mean and larger variations at higher resistance states. The security metrics for the PUF are experimentally characterized using a selection scheme with $m = 5$ rows and $n = 2$ columns and three different voltages V_B: 200 mV, 400 mV, and 600 mV. (According to Eq. (2), for this case, C_{MAX} = 697k for each voltage bias.) The measurements show that increasing the voltage bias from 200 mV to 600 mV improves UF from already decent 49.5 ± 6.25% to nearly ideal 50.1 ± 6.26%; another PUF randomness metric, DF, is also close to ideal, being ~50 ± 6.25% for all cases. The better PUF metrics at higher voltages are attributed to the stronger nonlinearity in the device *I-V*s.

The reliability is measured using the worst-case 16k challenges (out of 384k) that resulted in the smallest current differential readout margins. The results show that BER improves substantially at higher biases, from 3.9 ± 1.8% at V_B = 200 mV to 1.22 ± 1.0% at V_B = 600 mV; this is partially attributed to the improved readout margins and signal-to-noise ratio of the differential current. The improvement in BER is even more significant, from 16.36 ± 3.1% at 200 mV to 5.93 ± 2.59% at 600 mV, for PUF operation under an elevated ambient temperature of 90 ^{O}C. The uniqueness is initially evaluated by measuring the Hamming distance between the keys generated by pairs of PUF instances that are implemented by varying applied voltages V_B without re-tuning the device conductances. Not surprisingly, the maximum UQ of 44.8 ± 6.9% is achieved between the PUFs with the smallest applied voltage (200 mV) and the largest one (600 mV). This is quite natural because variations in nonlinear *I-V*s, are more prominent at higher biases, resulting in a non-monotonic redistribution of sneak

path currents. Then, characterization of the uniqueness between differently programmed crossbar devices also indicates close to the ideal 50% mean for all studied cases, with the smallest variance, 1.9%, at the largest bias voltage of 600 mV.

The major limitations of this demonstration are the shallow size of the used crossbar and the utilization of a complicated optimization algorithm to pre-compute the conductance of memristive devices. In the following, we propose several architectures, which address these limitations.

4.1. RX-PUF Design

RX-PUF [76] addresses these issues by harnessing an auxiliary computed bit in a two-cycle readout scheme. As a proof of concept, we have prototyped a 600 kb challenge-response pair PUF using 250 nm half-pitch 20×20 crossbar arrays of passively integrated Pt/Al_2O_3/TiO_{2-x}/Pt RRAM devices (Fig. 68a). The distribution of switching voltages, shown in Fig. 68b, is uniform enough to allow high-precision individual tuning of the devices in the crossbar. Similar to the original design, the sensed currents on each column are sums of the currents via selected *n* devices and the sneak path currents that are determined by the states of all floated devices. A different selection of rows and columns results in the redistribution of sneak-path currents, which is hard to predict or model due to *I-V* nonlinearities and their process-induced device-to-device variations. Here, we show that simple, normally distributed conductance of devices could generate high entropy keys without the need for using a tuning optimization procedure. Specifically, a two-step scheme is implemented using the circuit shown in Fig. 68c,d. First, the auxiliary sense amplifier (SA) sh, hardwired to the first and last columns, generates an “AUX” bit by comparing input currents in these lines. In the next step, a second output bit is generated by the main SA, which serves all but the mentioned two

columns. This bit is then XORed with AUX bit to produce the final response bit. In the implemented prototype, n = 5 (out of N = 20 total), while m = 2 (out of M = 20-2=18 total).

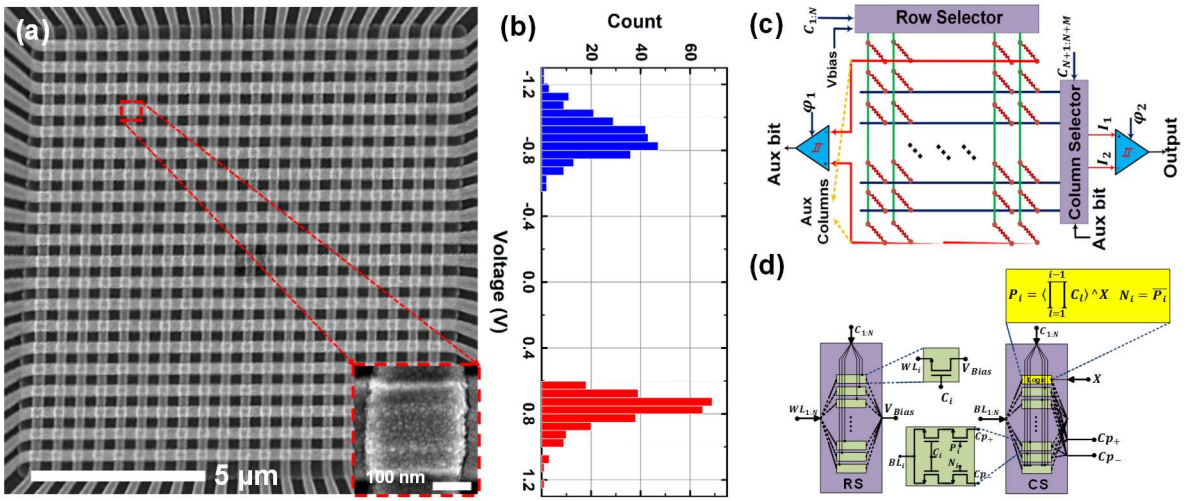


Fig. 68:(a) SEM micrograph of the 20×20 passive bilayer sputtered TiO_x and ALD-grown Al_2O_3 crossbar (scalebar: 5 µm). The inset shows a single crosspoint (scalebar: 100 nm). (b) histogram of SET and RESET switching voltage threshold distributions for crosspoint devices. (c) RX-PUF architecture and (b) a compact design of column/row selectors. The input is encoded by N+M bits, with 1s for the selected rows/columns.

Fig. 69a shows the distribution of the programmed conductances for the studied PUF instance. The implemented tuning is very crude (with conductances of ~ 20% devices significantly lower than the desired values) to show that the circuit works fine with an imbalance conductance distribution. We then collected output currents (Fig. 69b-c) and the corresponding responses (Fig. 69d) on all 600 kb CRPs. The measurement shows that the average uniformity of the demonstrated PUF due to the imperfect tuning and abandoning the balancing algorithm is 51.12%, which is reduced to the near-ideal 50.04% after harnessing the auxiliary bit. The measured BER is 0.7% at room temperature and ≤ 5.3% at 100°C, even without using any error correction methods.

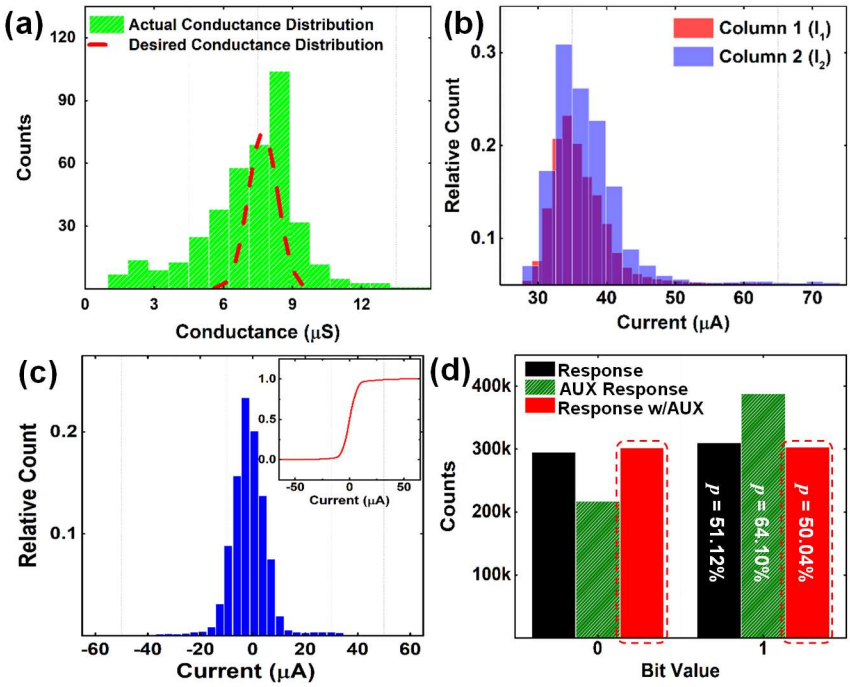


Fig. 69:(a) Histogram and (b) map of cells' conductances (at 0.3 V) for the demonstrated PUF. (b) Common-mode and (c) differential distributions of output currents sensed by main SA over the 600 kb responses. The inset shows ecdf. (d) The distribution of PUF response (at 0.3 V @ 25°C) with and without the AUX approach.

The measured inter-HD for 64-bit and 128-bit keys (Fig. 70a) shows near-ideal values (the average is ~50.0% in both cases), while the detailed analysis of such keys showed almost no correlations between the input bitstream and the output response for the AUX approach (Fig. 70b). The data clearly shows the benefit of the AUX approach, which allows reducing output bias due to imperfect tuning, and successfully passes the NIST randomness test (Fig. 70c). The resilience against machine learning attacks is assessed with a 40×500×500×1 MLP classifier. The classifier was trained on a subset of a specific size of the observed CRPs and then tested on another mutually exclusive observed set. Even with a large fraction of the training data, the classifier predicted output with close to ideal 50% accuracy for the PUF with the AUX approach (Fig. 70d). We expect that a moderate increase in crossbar array size may exponentially improve resilience even against modeling with very deep classifiers, while scaling-down device feature sizes would lead to sub-fJ operation.

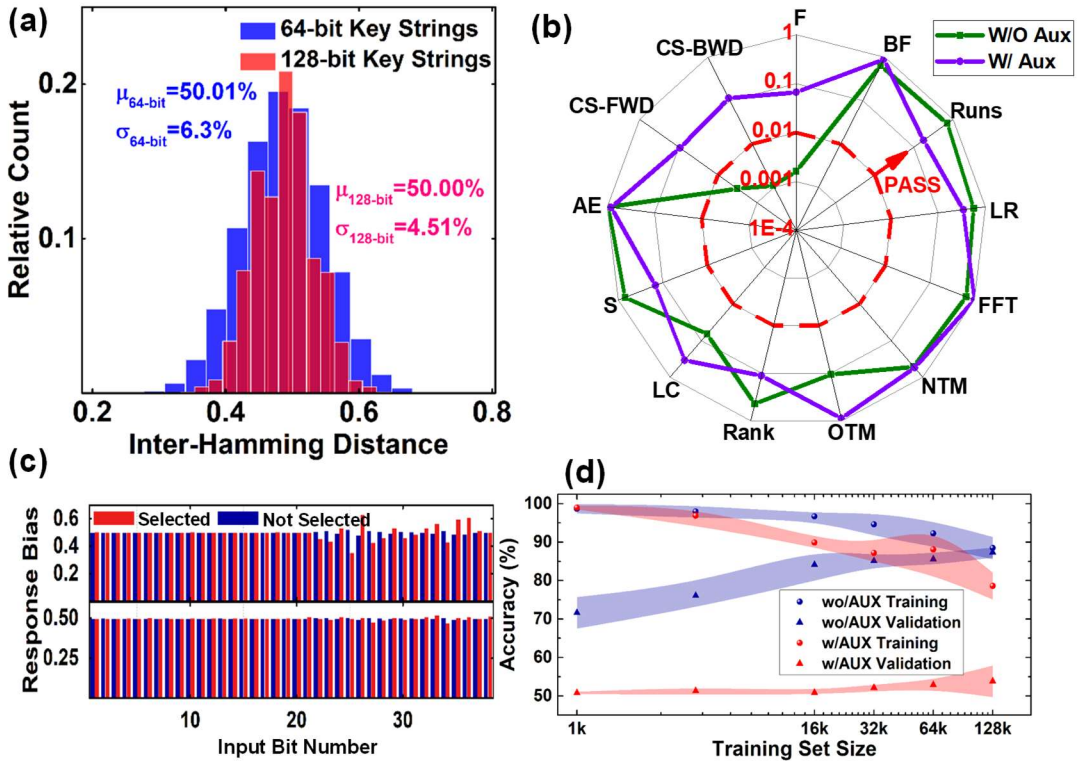


Fig. 70:(a) Inter-hamming distance (at 0.3 V @ 25°C) for 64 and 128-bit keys formed by grouping sequentially generated 1-bit responses. (b) Results of NIST randomness tests using measured 600kb responses for the PUF implementation with and w/o AUX. (c) Input-output correlations: Each column shows PUF output (at 0.3 V @ 25°C), averaged over all CRPs with the same fixed value of specific input bit. The bottom / top panel shows implementation with / w.o. AUX approach. (d) Modeling attack by MLP network, tested on 5k validation set, as a function of training set size (in bits). The training was performed using a gradient descent method with momentum. Symbols show average prediction accuracy, while the line thicknesses are drawn according to the max and min values obtained over 5 runs.

4.2. Ultra-Low Power VRPUF

Here, we present a novel design of physical unclonable function, called VRPUF, and prototype it using 4K-ReRAM passive crossbar circuits fabricated with a CMOS-compatible process, suitable for the back-end-of-line (BEOL) integration [211]. The VRPUF architecture utilizes intrinsic process-induced variations in as-fabricated (unformed) crossbars circuits to extract cryptographic keys, eliminating the need for forming and programming operations and their corresponding peripheral circuitry. The virgin-state operation reduces the power consumption due to the lower conductance state of individual devices, further improving the

power efficiency. The intrinsic variations in inter-device *I-V* nonlinearities, the main source of entropy in VRPUF response, is also significantly (~10×) more pronounced in as-fabricated devices as compared to formed devices further improving the VRPUF response unpredictability and attack resilience.

4.2.1. VRPUF Design

The major novelty of VRPUF design is the utilization of post-fabrication virgin-state crossbar circuits, whose conductance distribution (Fig. 71a) provides a rich entropy source for cryptographic secret key generation. This randomness is entirely intrinsic, unlike the previous ReRAM-based PUFs in which the devices must be formed and tuned to a narrow conductance distribution. The elimination of programming/forming steps has two advantages: (1) it simplifies peripheral circuits resulting in up to a ~4× improvement in area efficiency, and (2) dramatically limits the options for information leakage because adversaries are not physically capable of measuring the state of each device, individually. Furthermore, the statistical correlation between the pristine conductance states of crossbar circuits fabricated in the same process is negligible. Similar to the previous work, the proposed PUF design takes advantage of the nonlinear *I-V* characteristics and device-to-device variations in resistive memories to implement Shannon's confusion paradigm. The *I-V* nonlinearity and its variations are higher at larger biases (Fig. 71c), with the average nonlinearity at 0.4 V is ~80%, which is ~20× larger than that of a formed 50 kΩ device.

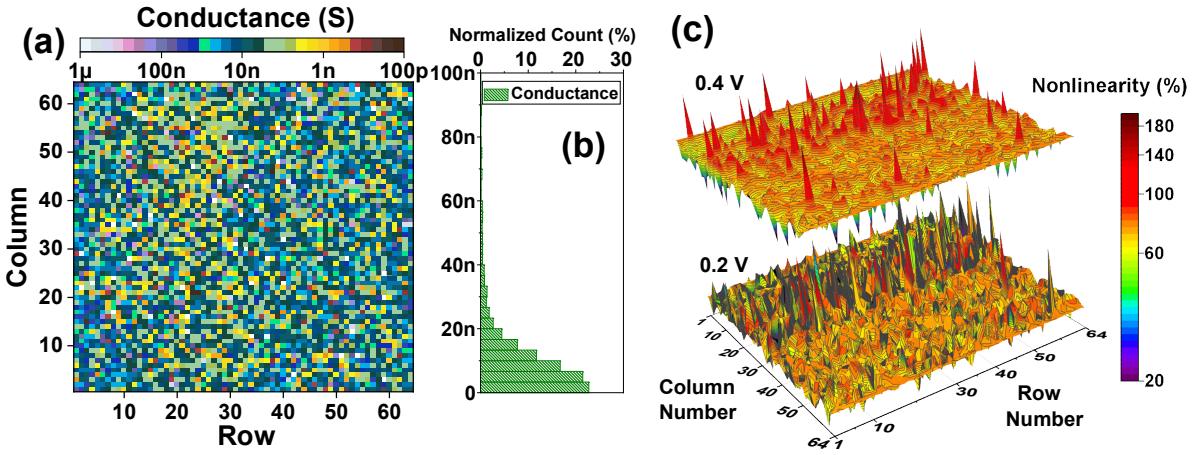


Fig. 71: (a) Map of cells' conductances measured at 0.4 V and room temperature (RT) for the demonstrated PUF and (b) its corresponding distribution. (c) Measured nonlinearity factor (i.e., $(|1 - G/G_0|) \times 100)$) of the demonstrated PUF for two biasing conditions at room temperature corroborating the presence of strongly nonlinear computing elements in the present design. (G_0 is defied at 0.1 V.)

One of the most serious concerns in ReRAM-based strong PUFs is outlier devices with large conductance, which create a bias in the response and strong input-output correlations. For example, the conductance distribution range in [211] spans through 5 orders of magnitude. Fig. 72 shows how this problem (e.g., due to unwanted forming of the devices by electrostatic discharge during wire-bonding or integration) is addressed in VRPUF's architecture. Specifically, a 192-bit length challenge is applied to the crossbar, from which 64 bits are used for row selection, and the rest is utilized for column selection in two cycles. Here, a single-bit response is generated after 4 cycles. In the first two cycles, bottom (top) lines are operated as columns (rows), two bits are generated and XORed. This provides us with the opportunity to read the current from both sides and possibly generates more entropy in the output. In the last two cycles, top (bottom) lines are operated as rows (columns) and the resultant two bits are XORed likewise. The final response bit is the result of XORing the two generated bits.

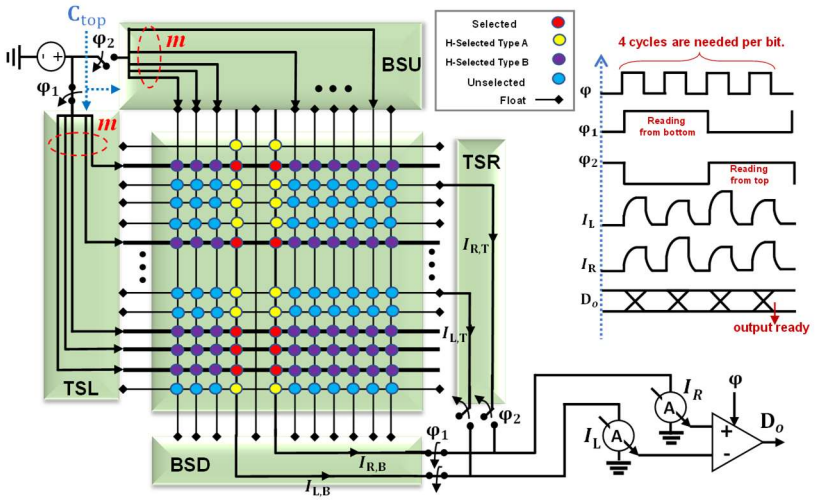


Fig. 72: The timing diagram and proposed PUF architecture, including a 64×64 crossbar, selectors (top, bottom, right, left), and a comparator. After 4 cycles, the final response is generated by XORing the previously readout bits from the bottom (2 bits) and top (2 bits).

4.2.2. VRPUF Experimental Results

The distributions of the common-mode and differential-mode currents sensed by the amplifier are shown in Fig. 73a Sub-μA common-mode currents enable extremely low-power consumption at 0.4 V, which is quite unusual for such a large-scale crossbar circuit. In fact, power consumption is ~ 100× less compared to previous designs adjusted to similar crossbar circuit complexity. The tight differential-mode current distribution stems from a uniform fabrication process, which has resulted in a very narrow conductance distribution. In general, a robust PUF should have negligible correlations between the input and output bits. Fig. 73b confirms that such correlations are insignificant in our design, which is in part due to the utilization of the multistep selection scheme in which the readout is performed on both top and bottom electrodes.

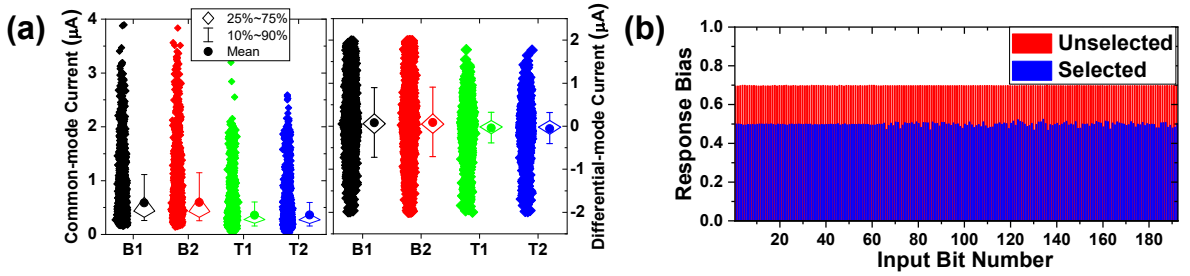


Fig. 73:The measured common-mode and differential current distributions of readout currents from the top and bottom electrodes in the first (B1, T1) and second (B2, T2) cycles for 2k challenges @ 0.4V and RT. (b) Measured input-output correlation between 110k CRPs at 0.4V and RT. Red shifted by +0.2 for clarity.

To further verify PUF robustness, we have performed extensive sets of simulations of resiliency towards machine learning attacks. Particularly, we have used a 192×100×100×100×1 MLP and online packages utilized in previous works [212], such as LIBSVM and LIBLINEAR. Training is performed on 95% of the data, among which 20% is used for cross-validation and the remaining mutually exclusive 5% CRPs are used for testing. The test accuracy for all networks is close to the ideal 50% prediction accuracy (which means the network can not predict the output response). We also tested a long short-term memory (LSTM) network consisting of two 128-unit LSTM layers (with rectified linear activation function) and two fully connected layers (with sigmoid activation function) based on the encoding scheme in [14] and observed that the predictability of the implemented network is ~50% for 32, 64, and 128-bit input sequences. The bitstreams generated by VRPUF pass all relevant NIST tests. We have also measured the reliability of the proposed PUF across a wide range of temperature and voltage deviations from the nominal condition. Due to the extremely low-power operation of the circuit, the differential-mode currents are sometimes comparable to the noise floor. This results in 3.73% unstable bits at the nominal conditions, which is still much better than the 16% in SRAM, ~34% in latch-based PUFs, and 30% in the hybrid circuit weak PUFs reported in [212,213]. Moreover, the proposed design can be classified as strong

PUF, and given its large CRP capacity (~10^{25}), discarding the unstable bits should not impact the security metrics. Masking unstable bits will reduce the worst-case bit-error rate at +10% voltage deviation by ~1.5×. The worst-case BER at 85°C is 9.5%. The uniformity distributions of measured 64 and 128-bit keys indicate near ideal response bias of VRPUF (Fig. 74a). As illustrated in Fig. 74b-c, the average diffuseness, and uniqueness of 64-bit keys generated by VRPUF instances are 49.96% and 50.03%, respectively.

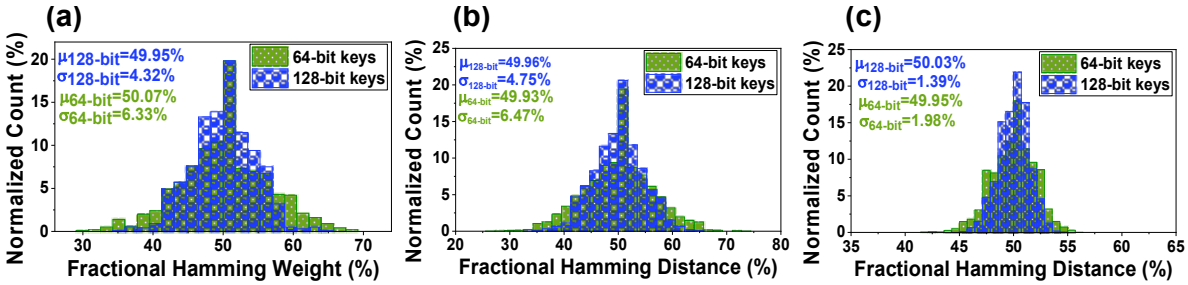


Fig. 74:(a) Uniformity: fractional Hamming weight distribution computed based on 1k randomly 64-bit and 128-bit generated keys at 0.4 V and RT. (b) Diffuseness: fractional Hamming distance distribution computed based on 1k randomly generated keys measured at 0.4 V and RT. (c) Uniqueness: fractional Hamming distance distribution computed based on 1k randomly generated keys of two PUF instances measured at 0.4 V and RT.

In summary, VRPUF operation is experimentally verified using unformed 64×64 crossbar circuits with passively-integrated ReRAM devices, fabricated with a novel CMOS-compatible etch-down patterning process. The results showed that the proposed circuits are ~4× denser, consume ~100× less power, and feature more robust security metrics compared to the previous work. Notably, VRPUF design does not require forming/ switching of crossbar devices, and hence using resistive memory is not essential (though practical for fabrication flows with the BEOL ReRAM process). The simpler device functionality broadens implementation options, which could ultimately lead to more cost-efficient CMOS integration. The only drawback of this design is the large BER at elevated temperatures, which is solved using a key-booking scheme as discussed in the next section.

4.3. Low BER VRPUF

We experimentally demonstrate a strong PUF circuit featuring $>2^{80}$ challenge-response pair capacity, < 1.4% worst-case BER [214]. The key contributions of this work are a novel leakage injection approach using an electrically isolated portion of the crossbar array, which boosts PUF's robustness, and a key-booking scheme, which dramatically improves reliability across a wide temperature range of operation and further increases PUF circuit density by reducing error-correcting overheads. A similar idea is previously suggested in the context of SRAM-based weak PUFs [215]. The main difference is that Ref. 6 work uses resource-hungry reverse fuzzy extractor and preprocessing techniques, such as majority voting and bit masking for error correction. On the other hand, our approach dramatically reduces raw BER without discarding [216], masking CRPs [217], and digital fine-tuning [218].

The studied baseline PUF architecture is also based on the virgin-state fixed-resistance passive crossbar circuits, which do not require programming circuitry and ensure very low-power secure operation by utilizing highly resistive crosspoint junctions. The median resistance of devices in the demonstrated PUF is ~22 MΩ. Fig. 75 shows the implemented strong PUF topology. A single bit response per 128-bit challenge is generated in two cycles. Specifically, the first 64 bits of a challenge uniquely determines sets of *m* selected rows that are grounded with CMOS peripheral circuits. The remaining two 32-bit sets of a challenge determine the two selected electrodes. The unselected electrodes are kept floating to circulate sneak-path current in the array, hence complicating the mapping. In each of the two cycles, currents flowing into the selected columns are compared using the dynamic current comparator to produce a single bit. The two bits are then XORed to generate one bit of a response. The total number of CRPs is $\sum\binom{64}{m}\binom{32}{2}$, which is $>2^{80}$ for $m = 16 \div 48$. Here,

we isolate 32 columns from the main circuit to ensure the consistent contribution of leakage current in the outputs, which further complicates the functional response. Due to the introduced leakage, when a single row is selected, the readout current is >10× larger than the expected range of values on average.

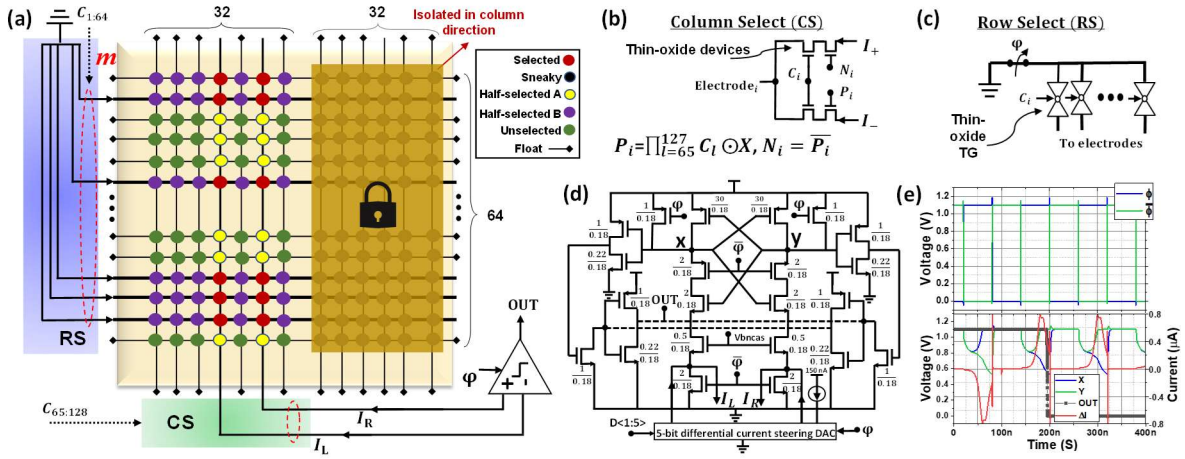


Fig. 75: The proposed PUF design. (a) Top-level diagram showing row/column selection, current comparator, and a 64×64 crossbar array circuit. (b, c) The design of (b) row select (RS) and (c) column selection (CS) circuits. Note the absence of large-area thick-oxide programming switches, which are not needed in the proposed PUF design due to using virgin-state (i.e., as-fabricated) RRAM devices. (d) Current comparator and (e) its simulated timing diagram.

4.3.1. Experimental Results

Due to the tunneling charge transport mechanism at the high-resistance virgin state, the devices' *I-V* characteristics are extremely nonlinear. More importantly, Fig. 76a shows that the average nonlinearity and its variations increase at larger biases (more than twofold increase in average and sevenfold in standard deviation when the bias voltage is increased from 0.15 V to 0.4 V), which is an important feature for improving robustness against machine learning attacks, as discussed earlier. Finally, the devices show excellent retention characteristics with no significant change in the conductance observed after baking the chip at 85 ℃ for >20 hours

and continuously measuring the conductance of the devices at 0.4 V (Fig. 76c). Such features enable the design of a reliable, strong PUF.

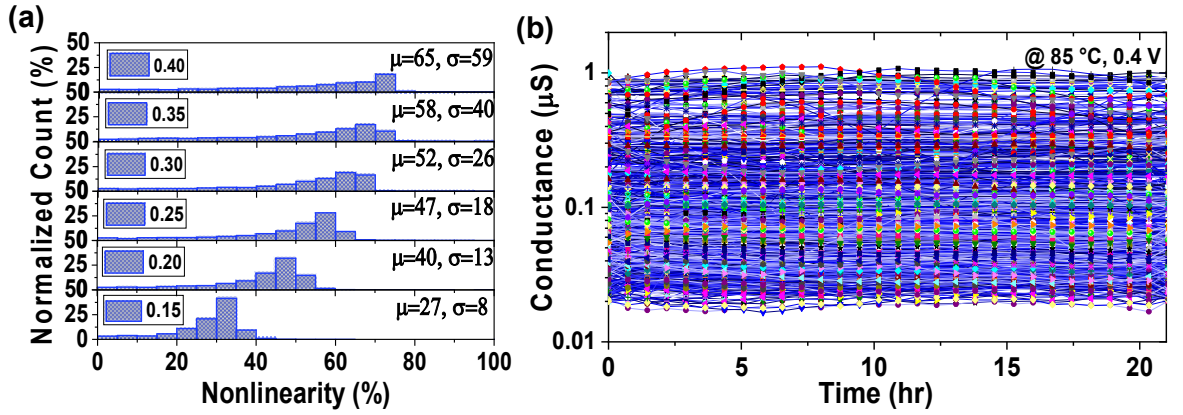


Fig. 76: ReRAM crossbar circuit characterization results. (a) The measured distributions of nonlinearity, defined as $(|G_0-G(V_{bias})|/G_0)\times 100$, and its variations for several values of Vbias. G_0 is the measured conductance at 0.1 V. (b) Accelerated retention test results at 85 °C for 1000 randomly selected devices.

We prototyped more than half a million CRPs using the demonstrated crossbar with randomly applied challenges from the CRP space. The symmetric distribution of differential-mode currents and relatively low common-mode response indicate high uniformity and low-power operation. The typical speckle pattern of 64-bit keys generated by the proposed PUF supports the high quality of generated bits (Fig. 77b). More quantitative, statistical analysis of >500,000 generated bits shows almost no direct correlation between inputs and output. Indeed, Fig. 77c shows that by selecting different row/column electrodes, the average response is always close to 50%, indicating no direct bias due to the selection of specific rows or columns. As shown in Fig. 77d, near-ideal uniformity, diffuseness, and uniqueness for 64-bit and 128-bit keys are achieved with the proposed PUF.

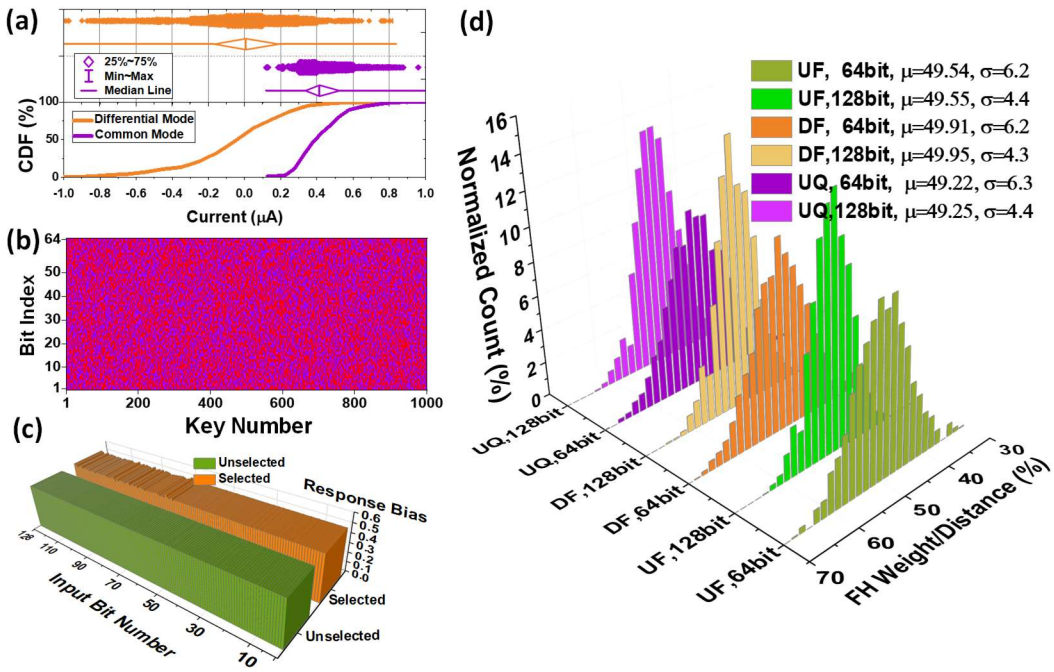


Fig. 77: General PUF characteristics. (a) Common-mode and differential distributions of output currents (top panel) and corresponding CDF (bottom panel) over 15 kb responses measured at 0.1 V and room temperature. (b) The speckle pattern for 1,000 64-bit keys. Blue / red shows '1'/ '0' responses. (c) Input-output correlations. Each column shows PUF output (at 0.4 V, @ 25 °C), averaged over outputs of all (500k) measured CRPs with the same fixed value of specific input bit. (d) Uniformity (UF), diffuseness (DF), and uniqueness (UQ), measured at 0.4 V and 85 °C. Fractional Hamming (FH) weight/distance distributions are computed based on 4k keys. All keys are formed by grouping generated 1-bit responses from randomly ordered challenges.

Unlike CMOS PUFs in which the response instability is a complex function of temperature, the output currents in the proposed PUF are semi-linearly dependent on the temperature with different slopes due to device-to-device and state-dependent variations. This is illustrated in Fig. 78a, which shows the output current as a function of temperature for 100 randomly picked CRPs. Such dependency results in a large BER at elevated temperatures, as shown in Fig. 78b (which is still better or comparable to native, i.e., uncorrected, BER of CMOS strong PUFs). To mitigate this issue, we propose a keybooking technique that takes advantage of the monotonic current-temperature relationship. Specifically, an error (a bit-flip) (if it happens) only occurs once in the entire temperature range. By storing the golden key at

multiple temperatures during the enrollment and retrieving the response at the closest to the operating temperature during authentication, we overcome this issue. For example, if golden keys are stored in three temperatures, namely 25℃, 45℃, 65℃ during enrollment (case KB3), we achieve intrinsic ~1.4% and ~0.7% errors for 0.1 V and 0.4 V biases across the whole temperature range (Fig. 78b). (Note that such BERs are largely dominated by noise, justifying the use of only a few reference temperatures during enrolment.) Hence, no on-chip post-processing and its significant overhead or CRP loss are imposed in our approach. In case the server has accurate information on the ambient temperature of the chip containing the PUF circuit, the overhead of the proposed technique is negligible – just an increase of the stored CRP table size on the server. Otherwise, the additional minor overhead is the addition of an on-chip temperature sensor, e.g., low-cost sensors available in IoT devices. Such a sensor would be used for informing the server about the chip temperature in both enrollment and authentication stages.

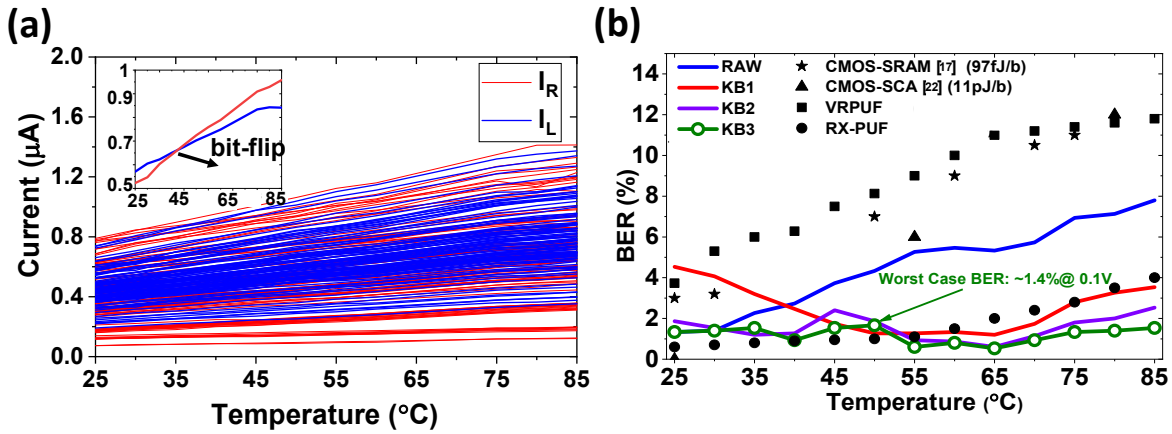


Fig. 78: General PUF characteristics. (a) Common-mode and differential distributions of output currents (top panel) and corresponding CDF (bottom panel) over 15 kb responses measured at 0.1 V and room temperature. (b) The speckle pattern for 1,000 64-bit keys. Blue / red shows '1'/ '0' responses. (c) Input-output correlations. Each column shows PUF output (at 0.4 V, @ 25 °C), averaged over outputs of all (500k) measured CRPs with the same fixed value of specific input bit. (d) Uniformity (UF), diffuseness (DF), and uniqueness (UQ), measured at 0.4 V and 85 °C. Fractional Hamming (FH) weight/distance distributions are

computed based on 4k keys. All keys are formed by grouping generated 1-bit responses from randomly ordered challenges.

The statistical properties of > 500k generated response bits are tested using the NIST test suite. The average p-value on all tests is well above the minimum pass value. The resilience against the machine learning attacks is first studied by modeling PUF with a 128×500×100×20×1 MLP. Specifically, the input challenge is applied as a binary 128-bit vector to the input layer of the perceptron, while a predicted single bit response is produced by the output layer of the network. The model is trained on a specific subset of measured CRPs using gradient descent with momentum and with a manually found quasi-optimal learning rate of 0.4. The testing is conducted on another CRP subset, which is mutually exclusive with the training subset. The test accuracy for all the cases is close to the ideal 50% prediction accuracy, even when increasing the training set size to ~5×10^5 CRPs. This means that the network, i.e., the attacker's model of the PUF circuit, always fails to make a meaningful prediction. The other machine learning approaches - LIBSVM and LIBLINEAR used in [212] also resulted in unsuccessful attacks with ~50% prediction accuracy, which corroborates the resiliency of the demonstrated PUF towards machine learning attacks. Fig. 79 shows the fabricated crossbar circuit and the experimental setup. Table 5 summarizes the measurement results and compares our design with previously reported PUFs based on CMOS and emerging technologies.

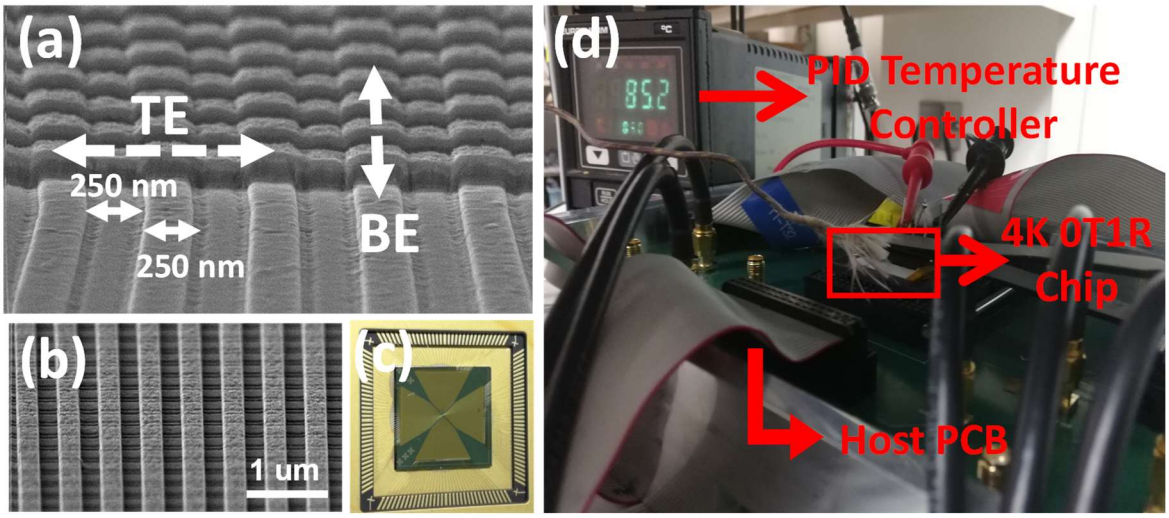


Fig. 79: Experimental setup. (a) Oblique-view and (b) top-view SEM images. The photos of (c) wire-bonded fabricated 64×64 crossbar circuit with bottom electrodes (BE) and top electrodes (TE) directions highlighted, and (d) the setup with the mounted circuits during BER measurements.

Table 5: Comparison with previous works (LL = LIBLINEAR package, LS = LIBSVM package, * at 0.1 V/ 0.4 V bias)

	ISSCC'15 [7]	ISSCC'15 [13]	VLSI'17 [3]	VLSI'17 [8]	VLSI'18 [4]	IEDM'19 [5]	DAC'19 [11]	This work
Technology	22 nm CMOS	65 nm CMOS	28 nm CMOS	130 nm CMOS	250nm RRAM	250nm RRAM	55 nm eFlash	250 nm RRAM
Demo Complexity	Crosscouple, 9581F^2	SA PUF, 12000 F^2	64×64 SRAM, ~ 194 F^2	SCA PUF	20×20, ~4 × 4 F^2 0T1R	64×64, 4 F^2 0T1R	5x 10x10, ~120 F^2	64×64, 4 F^2 0T1R
Pre-configuration	Intrinsic	Intrinsic	Intrinsic	Intrinsic	Preprogramming	Intrinsic	Preprogramming	Intrinsic
Capacity	-	~3× 10^3	~10^{11}	~3.7 × 10^{19}	~40 × 10^6	~7 × 10^{25}	~ 10^{211}	~2^{82}
BER @ 85 °C	~4.5 %	5 %	~ 12.5 %	~ 9 %	4.2 %	~ 11.5 %	~ 5 %	1.4% / 0.7%
NIST Test	Fail	Pass	Not Tested	Not Tested	Pass	Pass	Pass	Pass
ML Attack	Weak PUF	Weak PUF	LIBLIENAR, LIBSVM, 10k bits	LIBLIENAR, LIBSVM, NN,10k bits	MLP, 128k bits	MLP, LIBLINEAR, LIBSVM,LSTM, 80k bits	MLP, LIBLINEAR, LIBSVM, LSTM, 100k bits	MLP, LIBLINEAR, LIBSVM, 500k bits
Prediction Accuracy	-	-	~ 40%	~ 40%	~50%	~50%	~50%	~50%

4.4. *ChipSecure Design*

We exploit randomness in static *I-V* characteristics and reconfigurability of embedded flash memories to design efficient physically unclonable functions [219]. Leakage current and subthreshold slope variations, nonlinearity, nondeterministic tuning error, and sneak path current in the redesigned commercial flash memory arrays are exploited to create a unique digital fingerprint. A time-multiplexed architecture is designed to enhance the security and expand the challenge-response pair space to 10^{211}. Experimental results demonstrate 50.3% average uniformity, 49.99% average diffuseness, and native <5% bit error rate. The analysis of the measured data also shows strong resilience against machine learning attacks and the possibility for extremely energy-efficient operation.

Uncontrollable etching process leads to variations in the thicknesses of gate oxides and line-edge roughness, which in turn result in two major types of device-to-device variations in *I-V* characteristics: variations in subthreshold drain current with respect to gate voltage and drain voltage (due to drain-induced barrier lowering). The former can be decomposed into three separate factors, namely leakage, weak inversion, and higher-order effects. For example, Fig. 80a shows these three components by fitting log(*I*) expression of the measured subthreshold currents for 150 devices to a quadratic function of V_{CG}. Fig. 80b,c shows more details on the distribution of leakage currents (at V_{CG} = 0 V) for different values of V_{DS}, in particular highlighting a significant spread in leakage current distribution. (For all these experiments, memory cells have been tuned with better than 1% accuracy to eliminate the impact of tuning error in the measurements.) Programming the state of analog-grade nonvolatile memories is typically based on the write-verify algorithm, which has limited tuning accuracy (e.g., due to noise floor of the readout circuitry). Such tuning error can be used as an additional source of randomness. Fig. 80d, for example, shows the distribution of tuning error for 100 devices programmed randomly between 30 nA and 5 μA with a specified targeted accuracy. Fig. 80 provides extensive evidence of the variations and randomness in an array of programmed analog-grade flash memories. We exploit these entropy sources to design "ChipSecure" - a low-power and dense PUF instance with excellent uniformity and security. We first discuss the primitive block structure and detailed measurements and then follow with the top-level architecture and its results.

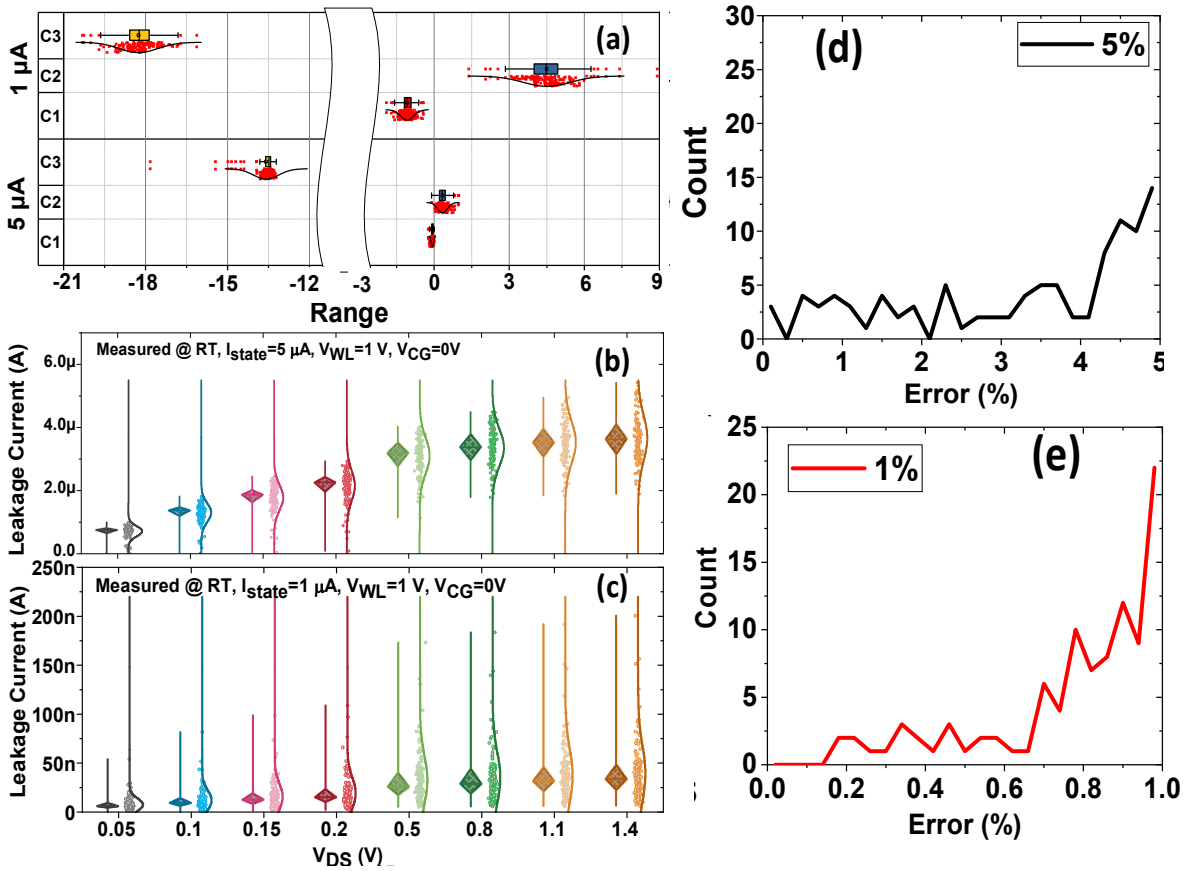


Fig. 80: Major entropy contributors in ChipSecure: (a) the results of fitting log(I) of the subthreshold drain currents to $C_1V_{CG}^2 + C_2V_{CG} + C_3$, highlighting significant variations of leakage component (C_3), weak-inversion slope (C_2), and higher-order nonlinearities (C_1). Box plots show 25%, 50%, and 75% quantiles. (b, c) Leakage current variations for the same 150 devices as a function of V_{DS} for two representative states (1 µA and 5 µA @ $V_{CG} = V_{WL} = V_{DS}$ =1 V), showing the nonlinear semi-exponential dependence of the leakage current on V_{DS}, with more prominent variations at larger biases. (d, e) Distribution of tuning accuracy for 100 devices programmed with (d) 5% and (e) 1% targeted accuracy to random states within the representative dynamic range. (Note that the shape of the distribution could be adjusted, if needed, with a more rigorous tuning procedure.

4.4.1. Primitive Block Design

The primitive building block is designed to exploit randomness in a network of nonlinear elements (Fig. 81). Guided by Shannon's security paradigm, the devices are operated at the deep subthreshold regime with exponential drain-source dependency so that the network is comprised of nonlinear computational elements, with all of them contributing to the output

response. The idea is similar to the memristive PUFs, in which sneak path currents are circulated in a crossbar of memristor arrays to build a compact security primitive. A primitive block consists of $N\times(M\text{-}N)$ array of floating-gate cells and peripheral (switching) circuitry for selecting cells and reading differential current (Fig. 81). The block is fed with an L-bit input challenge to generate a 1-bit output response. The input bits are partitioned into three groups: the first N bits of the challenge encode the selected word- (WLs) and control-gate (CG) lines; the next M-N bits are used to specify the selected bit-lines (BLs); while the remaining bits encode the selected source lines (SLs). (Note that each SL is shared between two rows of cells.) The selected CGs, WLs, and SLs are biased with $V_{CG,SEL}$, $V_{WL,SEL}$, and V_{SL}, respectively, while the selected BLs are grounded. Unselected CGs, BLs, and SLs are floated, while the unselected WLs are biased with $V_{WL,US}$. The response bit is generated by comparing the output selected currents. In our approach, cells in the array are categorized into four groups (inset of Fig. 81). For selected devices and half-selected type-A devices, SL is effectively a drain, while BL is a source. For the half-selected type-B devices, the current flows from BL to SL. The unselected devices can conduct current in either direction enabling circulation of sneak-path current in the array. Note that, in all cases, 1s/0s of an input bit-vector directly specify the position of the selected/unselected lines. This simplifies the implementation of peripheral circuitry by requiring only one MOS transistor switch per line. Current sensing and comparison are implemented similar to our previous design.

Our focus is on a 10×10 primitive block (i.e. with $N = 10$, $M = 20$, and $L = 25$), in which each challenge selects 5 WLs/CGs, 5 BLs, and 2 SLs. Furthermore, we consider Gaussian-distributed states (currents) of memory cells in the array. Finding the optimum distribution for cell currents during tuning and under nominal biasing conditions is future work. To

characterize block performance, we use the redesigned 10×10 memory arrays fabricated in 55 nm embedded CMOS. Keysight B1500A and B1530A tools and a custom-made switch matrix are utilized for characterization, programming, and measurements.

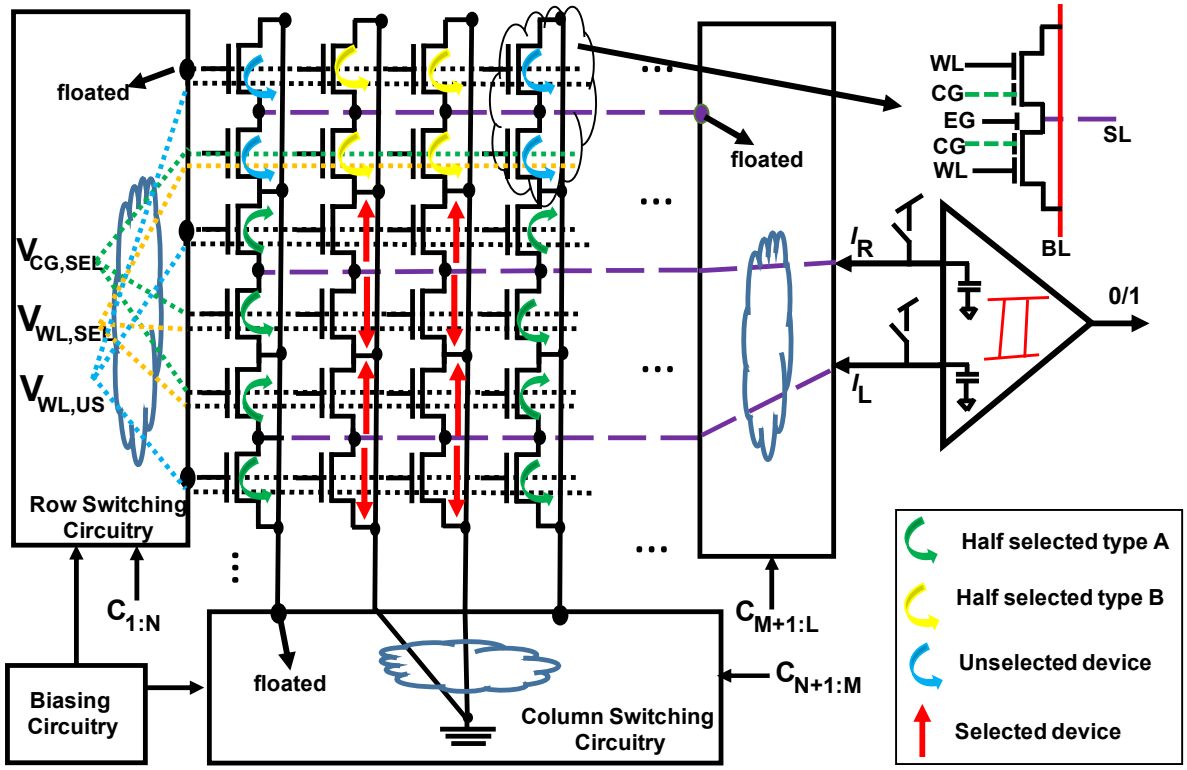


Fig. 81: ChipSecure primitive PUF block.

Fig. 82a shows an example of a current map after tuning the array with 10% accuracy to the randomly generated distribution with μ=500 nA and σ=150 nA. Due to the reconfigurability of our approach, a completely different map, i.e., a new fingerprint, is obtained after re-tuning the same physical array to a new distribution with μ=7.5 μA and σ=1.5 μA (Fig. 93b). Fig. 82c-d shows, respectively, the measured read-out current distribution (I_R and I_L) and their difference for the PUF instance (with $V_{WL,SEL}$ =1.25 V, $V_{WL,US}$=1.35 V, $V_{CG,SEL}$ = 0.3 V, and V_{SL} = 0.1 V) corresponding to Fig. 82a. The similar shapes of distributions indicate that there is no explicit bias in the output. The corresponding uniformity is 52.6%, which is close to the ideal, 50% value.

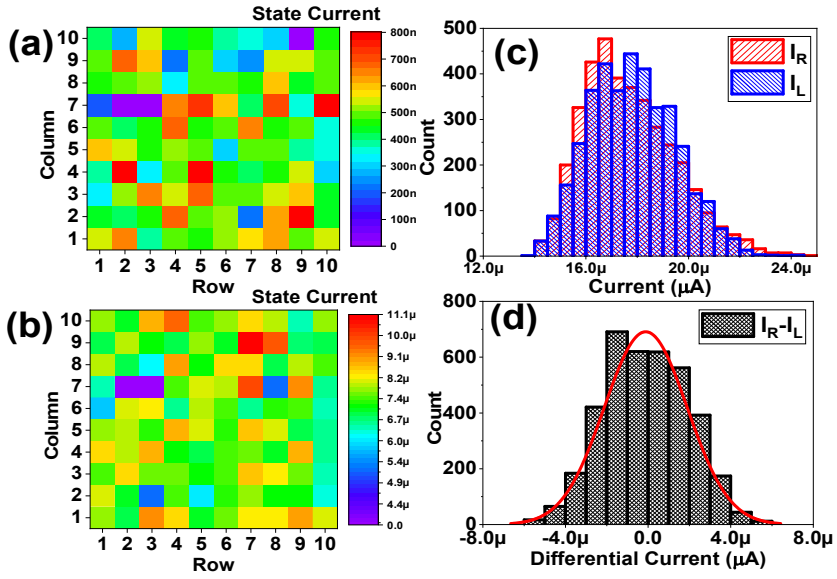


Fig. 82: Primitive block measurements at room temperature: (a,b) Two examples of the resultant map of conductance states in 10×10 array of cells, (c) the distribution of readout currents for 3k cases, and (d) the corresponding distribution of differential current.

The lack of a significant bias is also confirmed by the data in Fig. 83a, which shows that the output response is balanced with respect to the selected line in the array, i.e., the value of '1' at a certain position in the challenge bit-vector. Furthermore, we have measured the response uniformity of 12 different primitive blocks using 4 different silicon chips (Fig. 83b). For each primitive, we employed the same tuning procedure (Gaussian distribution with 10% targeted accuracy) but with different common-mode currents. Also, we have studied the sensitivity of uniformity metric to biasing conditions. This is performed by selecting appropriate V_{WL} from 0.65 V to 1.35 V, V_{SL} from 0.1 V to 0.5 V, and V_{CG} from 0.1 V to 0.5 V to match the selected common-mode currents for each instance. For each block, 4K randomly selected challenges have been applied, and the response is measured at room temperature. The experimental results show again close to 50% uniformity for the majority of the considered instances.

Perhaps, the most interesting result is the dependence of BER on the utilized common-mode current (Fig. 83c-e). This experiment is performed using 5 different block instances with

specified current-mode currents. Each primitive block is characterized by measuring responses to 1K challenges at different ambient temperatures and nominal voltage deviations. The results show that increasing temperature above the nominal 25 °C, at which devices are tuned, results in a semi-quadratic increase of BER, while the reliability is always improved by operating at higher bias currents. This is most likely due to the weaker temperature dependency at larger subthreshold currents. Indeed, the currents become almost independent of the temperature at some point in the strong inversion. Therefore, there is a clear trade-off between power consumption and BER, and, e.g., the desired operating point could be determined based on the power budget and BER requirements of the application. The same trend in BER is also observed with respect to the variations on the biased SL voltage (Fig. 83e), though the dependence is weaker.

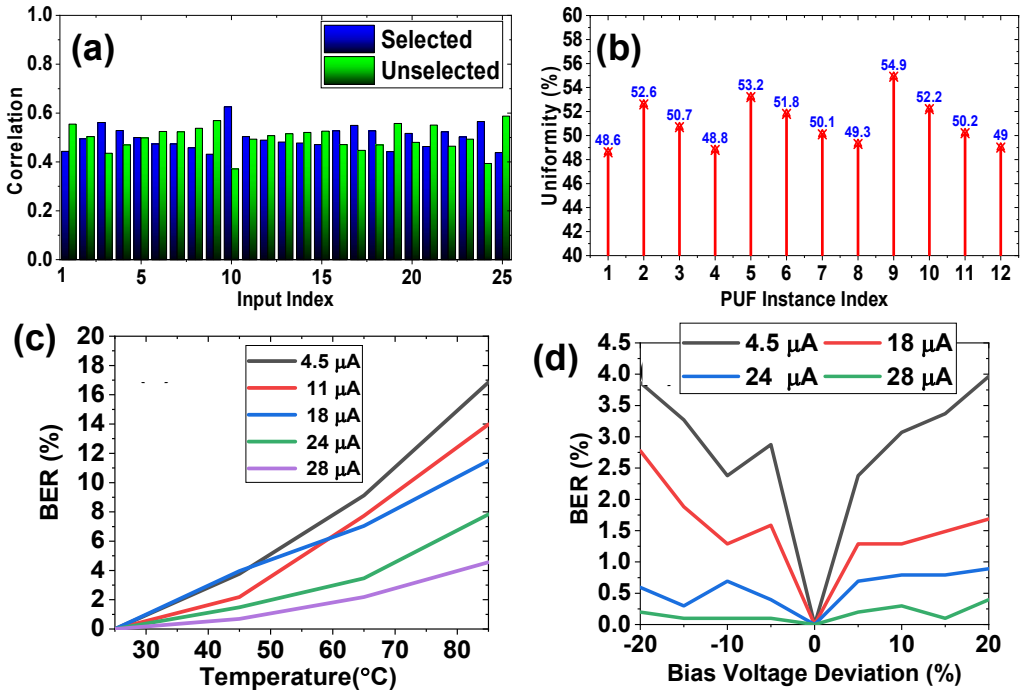


Fig. 83: Primitive block measurements: (a) Measured correlation (fraction of 1s in the response when particular bit at the input is selected) based on 4K random challenge-response pairs. (b) Response uniformity of 12 different primitive blocks obtained from reprogramming 4 different chips. (c) Measured BER as a function of (d) temperature at nominal SL readout

voltage for several common-mode readout currents and (e) bias voltage deviation for different common-mode currents at room temperature.

4.4.2. ChipSecure Architecture

The ChipSecure architecture consists of two layers of primitive blocks connected via a hidden shift register (HSR) (Fig. 84). An input challenge C is a 1010-bit vector. It is partitioned into subvectors C_1 C_2… C_{42}, such that C_i s with $1 \leq i \leq 40$ are 25-bit long, while C_is with $41 \leq i \leq 42$ are comprised of 5 bits. The C_is from the first group are used as an input to five primitive blocks ($P_{1\text{-}5}$) in the first layer. The remaining two subvectors are used to select SLs of the two primitive blocks ($P_{6,7}$) in the second layer. ChipSecure operation is time-multiplexed with the help of control signals φ_{L1}, φ_{L2}, and φ_C. Specifically, φ_{L1} controls the first layer to generate and fill in HSR with 40 response bits over 8 cycles of operation, with a total of 5 bits produced in each cycle by five corresponding primitive blocks (e.g., $D_{1\text{-}5}$ in the first cycle). Using φ_{L1}, HSR bits are applied to WL/CG and BL of $P_{6\text{-}7}$ to generate outputs R_1 and R_2. These outputs are then XORed to generate the final response (R_F) and reduce any undesirable bias. Assuming the delay of each block is t_d, a simple pipelining of the design would result in $1/8t_d$ (bit per second) throughput. We can achieve ~192.3 Mbps throughput by using a comparator that consumes 1.625 fJ per conversion with 1 nA sensitivity in 55 nm CMOS. The proposed time-multiplexing approach also enables a low area implementation with an enormous number of CRPs, which is a necessary condition for a secure strong PUF, at the expense of lower throughput and decreased energy-efficiency. Indeed, for the proposed scheme, the maximum number of distinct selections for $P_{1\text{-}6}$ is $S_P = \binom{10}{5}\binom{10}{5}\binom{5}{2}$. Note that all primitive blocks in the first layer contribute equally (4 bits) to the 20 bits needed in $P_{6,7}$. The total number of distinct selections is given by $\binom{S_P}{8}^5\binom{5}{2}^2 \approx 10^{211}$.

It is worth mentioning the benefits of using smaller arrays, which are essential for time-multiplexed implementation. First, smaller arrays are more efficient for utilizing the sneak path currents because leakage currents in larger arrays would be mostly controlled by a relatively small fraction of semi-selected cells. Second, an undesired stuck-on device in an array can bias the PUF response and potentially make the circuit vulnerable to probing attacks. Clearly, such stuck-on devices can be mitigated more efficiently when using smaller arrays with the time-multiplexed scheme.

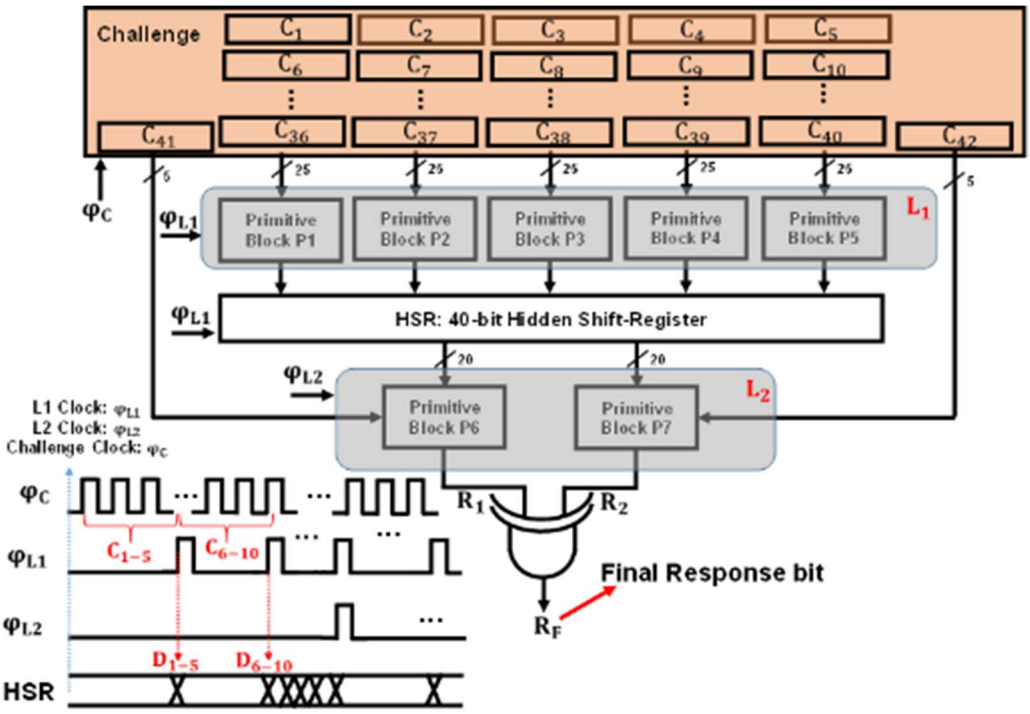


Fig. 84: The ChipSecure architecture and its timing diagram.

4.4.3. Measurement Results

The ChipSecure is characterized using the aforementioned primitive block design. The measured currents are used in the post-layout simulation setup of CMOS circuits. The fully integrated design occupies ~1.3×1.0 mm^2. It is dominated by low-voltage (0.3 mm^2) and high-voltage (0.1 mm^2) IOs and unused silicon (~0.9 mm^2). Active circuits, including programming

circuitry (4475 μm^2), flash memory array (235 μm^2), registers (19,250 μm^2), comparators (150 μm^2), and logic (110 μm^2) are very compact (total of 24,216 μm^2). Fig. 85a shows that the measured uniformity of HSR bits is near-ideal for $P_{1,2,4}$ blocks, though there is visible bias in $P_{3,5}$ responses. Despite that, the differential current distribution of $P_{6,7}$ looks symmetrical as shown in Fig. 85b for P_6. (Here, P_6 is tuned using 500 nA average state current and operated at $V_{WL,SEL}$=0.85 V, $V_{WL,US}$=0.9 V, $V_{CG,SEL}$=0.3 V, and V_{SL}=0.3 V.) Interestingly, the measured correlations, based on 100K challenge-response pairs, are much weaker than, as compared to those for a single primitive block (Fig. 96c). The randomness in the output response is also highlighted by the speckle pattern of 1K randomly selected 128-bit keys shown in Fig. 85d.

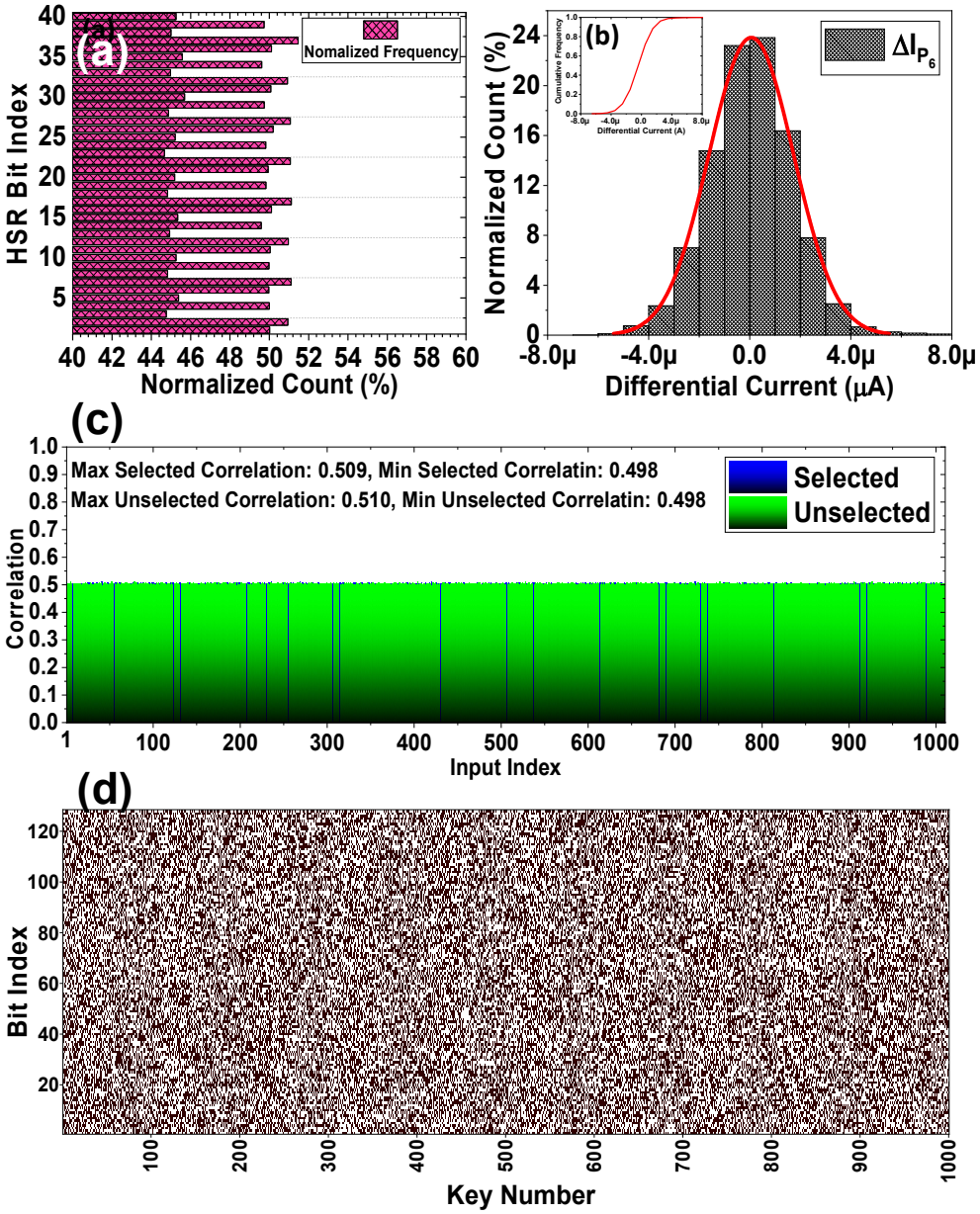


Fig. 85: ChipSecure measurement results: (a) Normalized Hamming weight of HSR bits over 100K applied challenges. (b) The differential current distribution of P_6, with the inset showing the corresponding CDF. (c) Measured correlation based on 100K random challenge-response pairs. (d) The speckle pattern for 1K 128-bit keys (black=’1’).

Fig. 86a shows the fractional Hamming weight distribution for 5K 64-bit and 128-bit keys generated based on R_1, R_2, and R_F. Based on these results, ChipSecure offers an average uniformity of 50.3%. (The measured uniformities for a larger set with 100K responses are 50.9%, 52.0%, and 50.3% for R_1, R_2, and R_F, respectively.) Fig. 86b shows near-optimal

results for diffuseness - the other important metric that evaluates the difference (Hamming distance) between unique keys generated by the same PUF under different challenges.

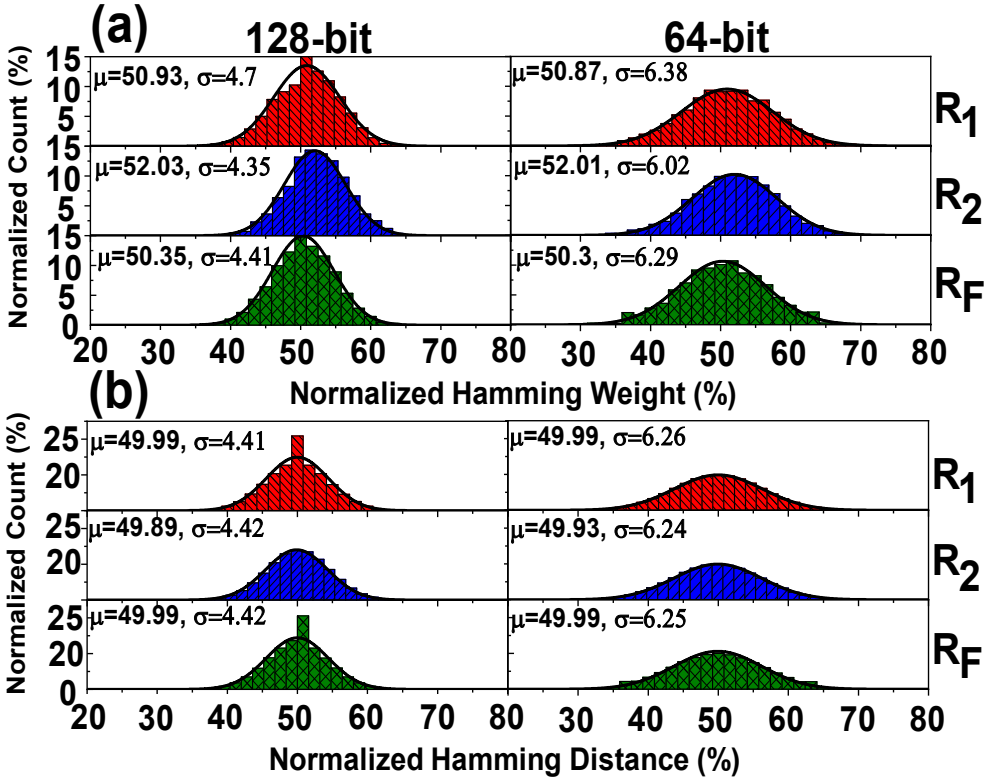


Fig. 86: (a) Fractional Hamming weight and (b) fractional Hamming distance distribution of R_1, R_2, and R_F. The results are computed based on (a) 5K and (b) 1K randomly-generated 64-bit and 128-bit keys.

As mentioned before, we can tune the devices to achieve a certain common-mode current and, subsequently, a desirable BER. Based on Fig. 83d, the native BER of <5% can be achieved with ~30 μA. In this case, we estimate the energy efficiency of 0.56 pJ/bit can be achieved with 88% /12% contributed by array/comparators. The bookkeeping scheme that is employed to improve the BER of VRPUF can be used in ChipSecure as well. Machine learning attack resiliency is assessed with a massive multilayer perceptron network (1010×100×100×100×1) with rectified linear activation function in the hidden layers and a sigmoid activation function in the output layer. We used RMSprop as an optimizer and a

learning rate of 0.001. 64% and 16% of CRPs are used for training and validating the network, respectively. The classifier was trained with a specific size subset of the observed challenge-response pairs and then tested on another mutually exclusive data. The test accuracy is close to the ideal 50% prediction accuracy even after using 120 Kbits for training the model. We also used both networks (LIBSVM and LIBLINEAR) introduced in [212], and for both, the validation accuracy is also close to 50%. The generated 100K bits also pass all relevant NIST tests.

Finally, in Ref. [220], we modify the ChipSecure architecture to design a lightweight, suitable for Internet of Things devices, integrated design of PUFand TRNG blocks. Shared silicon in designing the PUF and TRNG blocks results in a very compact and energy-efficient topology. High Shannon entropy is achieved in the experimentally generated TRNG bits. In addition, accelerated aging measurements indicate stable physical unclonable function response after 900 min of baking at 85 ◦C. Fig. 87 shows the fabricated array and experimental setup for BER and aging measurements.

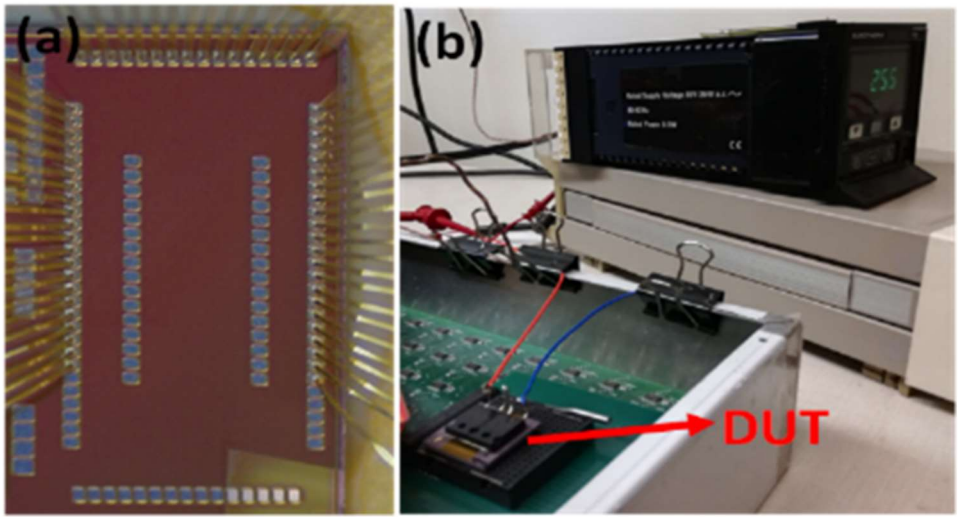


Fig. 87: Die photo of a primitive block fabricated in GF's 55nm CMOS and (b) measurement setup.

4.5. Summary and Future Works

In summary, we showed that the analog tuning and nonlinear conductance variations of memristors could be used to build memristor-based resilient, dense, fast, and energy-efficient physically unclonable functions. The initial design that was based on a 3D integrated 10×10 metal-oxide memristive crossbar circuit shows a near-ideal functional performance. To mitigate the shortcomings of this design, we proposed RX-PUF that takes advantage of a two-step readout scheme to avoid the need for a complex conductance tuning algorithm. Further, we present a novel architecture, called VRPUF, and prototype it using unformed 4K-ReRAM passive crossbar circuits. The architecture utilizes intrinsic process variations in as-fabricated crossbar circuits and allows for a massive number of challenge-response pairs (~10^{25}). In an improved design, the main limitation of VRPUF is mitigated by a key-booking scheme, which dramatically improves reliability across a wide temperature range of operation and further increases PUF circuit density by reducing error-correcting overheads. We also demonstrated ChipSecure, a PUF architecture that exploits randomness in the static *I-V* characteristics of eFlash memories. A two-layer time-multiplexed architecture enhances the security and expands the challenge-response pair space. Comprehensive characterization of the proposed PUF instances indicates clear advantages of these technologies over pure CMOS strong PUFs. To summarize, the experimental results from eFlash and memristive PUFs demonstrate exciting opportunities for using analog memories in designing energy-efficient and scalable hardware security primitives based on hardware roots of trust.

A near-term future is the fabrication of a CMOS-integrated memristive PUF. Our primary estimations show that the design of a PUF array that includes 16 32×32 double-sided blocks integrated with high-speed comparators, control logic, and shared scan register bank

consumes ~6.2 mm^2 in a 5M1P 180 nm CMOS technology. Note that 80% of this design is IOs and unused silicon. This massive chip has an enormous capacity of 1.5×10^{14} CRPs per block, which can time-multiplexed in a general-purpose fashion to produce gazillions of CRPs. Our simulation results indicate an impressive 1.6 Gbps state entropy generated, which is limited by IO speed and serialization input challenge bits. The design also has zero static power consumption in the core circuit, and we expect to achieve ~50 fJ/b energy efficiency.

Another future work is the fabrication of an integrated eFlash-based PUF chip in 55 nm CMOS. A preliminary design of this system is already completed. This PUF circuit features a time-multiplexed two-layer cryptographic engine with pipeline architecture that relies on process-induced variations, harnessed in the design of ChipSecure. The design has immunity toward various side-channel and fault-injection attacks, and it includes a large number of CRPs that are impossible to learn, extrapolate, and even compute in a lifetime. This novel architecture includes ~1.2M eFlash cells, 2346 registers, and 24 analog peripheries. The estimated processing speed is 2.1 μs for generating a 16-bit response stream, and the area of the chip excluding the bonding pads is estimated 3.4 mm^2.

References

[1] Zhang, W., *et al.* Neuro-inspired computing chips. *Nature Electronics* **3**, 2020.

[2] Alibart, F., E. Zamanidoost, and D. B. Strukov. Pattern classification by memristive crossbar circuits using ex situ and in situ training. *Nature communications* **4**, 2013.

[3] Mead, C. Neuromorphic electronic systems. *Proceedings of the IEEE* **78**, 1990.

[4] Indiveri, G., *et al.* Neuromorphic silicon neuron circuits. *Frontiers in neuroscience* **5**, 2011.

[5] Likharev, K. K., Hybrid CMOS/nanoelectronic circuits: Opportunities and challenges. *Journal of Nanoelectronics and Optoelectronics* **3**, 2008.

[6] Wang, Z., *et al*. Resistive switching materials for information processing. *Nature Reviews Materials* **5**, 2020.

[7] Hasler, J., and H. B. Marr. Finding a roadmap to achieve large neuromorphic hardware systems. *Frontiers in neuroscience* **7**, 2013.

[8] Mahmoodi, M. R., M. Prezioso, and D. B. Strukov. Versatile stochastic dot product circuits based on nonvolatile memories for high performance neurocomputing and neurooptimization. *Nature Communications* **10**, 2019.

[9] Prezioso, M., *et al.* Spike-timing-dependent plasticity learning of coincidence detection with passively integrated memristive circuits. *Nature Communications* **9**, 2018.

[10] Yi, W., *et al.* Biological plausibility and stochasticity in scalable VO 2 active memristor neurons. *Nature Communications* **9**, 2018.

[11] Cai, F., *et al.* Power-efficient combinatorial optimization using intrinsic noise in memristor Hopfield neural networks. *Nature Electronics* **3**, 2020.

[12] Kumar, S., J. P. Strachan, and R. S. Williams. Chaotic dynamics in nanoscale NbO 2 Mott memristors for analogue computing. *Nature* **548**, 2017.

[13] Wang, Z., *et al.* Reinforcement learning with analogue memristor arrays. Nature Electronics **2**, 2019.

[14] Nili, H., *et al.* Hardware-intrinsic security primitives enabled by analogue state and nonlinear conductance variations in integrated memristors. *Nature Electronics* **1**, 2018.

[15] Kim, J., *et al.* Nano-intrinsic true random number generation: A device to data study. *IEEE Transactions on Circuits and Systems I: Regular Papers* **66**, 2019.

[16] Bavandpour, M. *et al.* Mixed-signal neuromorphic inference accelerators: Recent results and future prospects. in *Proc. IEEE International Electron Device Meeting (IEDM)*, 2018.

[17] Bayat, F. M. *et al.* Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits. *Nature Communications* **9**, 2018.

[18] Mead, C. Analog VLSI and Neural Systems. Boston, MA, USA: Addison-Wesley, 1989.

[19] Sarpeshkar, R. Analog versus digital: Extrapolating from electronics to neurobiology. *Neural Comput.,* **10**, 1998.

[20] Suma, G., *et al.* A programmable and configurable mixed-mode FPAA SoC. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* **24**, 2016.

[21] Wong, H. P., and S. Salahuddin. Memory leads the way to better computing. *Nature nanotechnology* **10**, 2015.

[22] Ambrogio, S., *et al.* Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature* **558**, 2018

[23] Shi, Y., *et al.* Neuroinspired unsupervised learning and pruning with subquantum CBRAM arrays. *Nature communications* **9**, 2018.

[24] Prezioso, M., *et al.* Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **521**, 2015.

[25] Li, C., *et al.* Efficient and self-adaptive in-situ learning in multilayer memristor neural networks. *Nature communications* **9**, 2018.

[26] Adam, G. C., *et al.* 3-D memristor crossbars for analog and neuromorphic computing applications. *IEEE Transactions on Electron Devices* **64**, 2016.

[27] Johnson, M. *et al.* Google's multilingual neural machine translation system: Enabling zero-shot translation. *Trans. Assoc. Comput. Linguist.* **5**, 2017.

[28] https://nv-adlr.github.io/MegatronLM

[29] Shazeer1, N., *et al.* Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. Preprint at https://arxiv.org/abs/1701.06538., 2017.

[30] Mountcastle, V. The Cerebral Cortex, Harvard U. Press, 1998.

[31] Yu, S., *et al.* Binary neural network with 16 Mb RRAM macro chip for classification and online training. in *Proc*. *IEEE International Electron Device Meeting (IEDM)*, 2016.

[32] Yao, P., *et al.* Face classification using electronic synapses. *Nature Communications* **8**, 2017.

[33] Lin, P., *et al.* Three-dimensional memristor circuits as complex neural networks. *Nature Electronics* **3**, 2020.

[34] Yang, T. J., *et al.* Design considerations for efficient deep neural networks on processing-in-memory accelerators. in *Proc*. *IEEE International Electron Device Meeting (IEDM)*, 2019.

[35] Mishra, A., *et al.* WRPN: Wide reduced-precision networks. *arXiv*:*1709.01134,* 2017.

[36] Darabi, S., *et al.* BNN+: Improved binary network training. *arXiv:1812.11800,* 2018.

[37] Deguchi, J., *et al.* Can in-memory/analog accelerators be a silver bullet for energy-efficient inference? in *Proc*. *IEEE International Electron Device Meeting (IEDM)*, 2019.

[38] Alibart, F., *et al.* Pattern classification by memristive crossbar circuits using ex situ and in situ training. *Nature communications* **4**, 2013.

[39] https://www.sst.com/

[40] Guo, X., *et al.* Temperature-insensitive analog vector-by-matrix multiplier based on 55 nm NOR flash memory cells. in *Proc*. *IEEE Custom Integrated Circuits Conference (CICC)*, 2017.

[41] Bayat, F. Merrikh, *et al.* Redesigning commercial floating-gate memory for analog computing applications. in *Proc*. *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015.

[42] Burr, G. W. *et al.* Neuromorphic computing using non-volatile memory. *Advanced Physics* **2**, 2017

[43] Hoskins, B. D. and Strukov, D. B. Maximizing stoichiometry control in reactive sputter deposition of TiO_2. *J. Vac. Sci. Technol. A* **35**, 2017.

[44] Kaneko, A., *et al.* Sidewall transfer process and selective gate sidewall spacer formation technology for sub-15nm FinFET with elevated source/drain extension. in *Proc*. *IEEE International Electron Device Meeting (IEDM)*, 2005.

[45] Boixo, S., *et al.* Evidence for quantum annealing with more than one hundred qubits. *Nature Physics* **10**, 2014.

[46] Yamaoka, M., *et al.* A 20k-spin Ising chip to solve combinatorial optimization problems with CMOS annealing. *IEEE Journal of Solid-State Circuits* **51**, 2015.

[47] Shim, Y., *et al.* Ising computation based combinatorial optimization using spin-Hall effect (SHE) induced stochastic magnetization reversal. *Journal of Applied Physics* **121**, 2017.

[48] Inagaki, T. *et al.* A coherent Ising machine for 2000-node optimization problems. *Science* **354**, 2016.

[49] Kirkpatrick S. Optimization by simulated annealing: Quantitative studies. *Journal of statistical physics* **34**, 1984.

[50] Chen, L. and K. Aihara, Chaotic simulated annealing by a neural network model with transient chaos, *Neural networks* **8**, 1995.

[51] Cheemalavagu, S., *et al.* A probabilistic CMOS switch and its realization by exploiting noise. in: *Proc. IFIP International Conference on Very Large Scale Integration (VLSI-SoC'05)*, 2005.

[52] Sutton, B., *et al.,* Intrinsic optimization using stochastic nanomagnets. *Nature Scientific Reports* **7**, 2017.

[53] Ostwal, V., *et al.* Spin-torque devices with hard axis initialization as stochastic binary neurons. *Nature Scientific Reports* **8**, 2018.

[54] Tuma, T., *et al.* Stochastic phase-change neurons. *Nature Nanotechnology* **11**, 693–699, 2016.

[55] Gong, N., *et al.* Signal and noise extraction from analog memory elements for neuromorphic computing. *Nature Communications* **9**, 2018.

[56] Lin, Y., *et al.* Demonstration of generative adversarial network by intrinsic random noise of analog RRAM devices. in: *Proc. IEEE International Electron Devices Meeting (IEDM)*, 2018.

[57] Ambrogio, S., *et al.* Statistical fluctuations in HfOx resistive-switching memory: Part I - set/reset variability. *IEEE Trans. on Electron Devices* **61**, 2014.

[58] Gaba, S., *et al.* Stochastic memristive devices for computing and neuromorphic applications. *Nanoscale* **5**, 2013.

[59] Shin, J., *et al.* Hardware acceleration of simulated annealing of spin glass by RRAM crossbar array. in: *Proc. IEEE International Electron Devices Meeting (IEDM)*, 2018.

[60] Guo, X., *et al.* Modeling and experimental demonstration of a Hopfield network analog-to-digital converter with hybrid CMOS/memristor circuits. *Frontiers in neuroscience* **9**, 2015.

[61] Hu, S., *et al.* Associative memory realized by a reconfigurable memristive Hopfield neural network. *Nature Communications* **6**, 2015.

[62] Danial, L., *et al.* Two-terminal floating-gate transistors with a low-power memristive operation mode for analogue neuromorphic computing. *Nature Electronics* **2**, 2019.

[63] Jouppi, N., *et al.* In-datacenter performance analysis of a tensor processing unit. in *Proc. Int. Symp. Comput. Archit. (ISCA),* 2017.

[64] Lee, J., *et al.* UNPU: An energy-efficient deep neural network accelerator with fully variable weight bit precision. *IEEE J. Solid-State Circuits* **54**, 2019.

[65] Chen, Y., *et al.* Eyeriss: An energyefficient reconfigurable accelerator for deep convolutional neural networks. *IEEE J. Solid-State Circuits* **52**, 2017.

[66] Sze, V., *et al.* Efficient processing of deep neural networks: A tutorial and survey. *Proc. IEEE* **105**, 2017.

[67] Bavandpour, M. *et al.* aCortex: An Energy-Efficient Multipurpose Mixed-Signal Inference Accelerator. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits* **6**, 2020.

[68] Chen, W.-H., *et al.* A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors. in: *Proc.* 2018 IEEE *Int. Solid - State Circuits Conference(ISSCC),* 2018.

[69] Yao, P., *et al.* Fully hardware-implemented memristor convolutional neural network. *Nature* **577**, 2020.

[70] Guo, X., *et al.* Fast energy-efficient robust and reproducible mixed-signal neuromorphic classifier based on embedded NOR flash memory technology. in: *Proc. IEEE International Electron Devices Meeting (IEDM)*, 2017.

[71] Merolla, P. A., *et al.* A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science* **345**, 2014.

[72] Larimian, S., *et al.* Improving Machine Learning Attack Resiliency via Conductance Balancing in Memristive Strong PUFs. *SRC Techon,* 2020.

[73] Mahmoodi, M. R., *et al.* Experimental demonstrations of security primitives with nonvolatile memories. *IEEE Transactions on Electron Devices* **12**, 2019.

[74] Xi, X., *et al.* Fresh re-keying with strong PUFs: A new approach to side-channel security. in: *Proc. International Symposium on Hardware Oriented Security and Trust (HOST),* 2018.

[75] Adam, G., *et al.* Utilizing IV non-linearity and analog state variations in ReRAM-based security primitives. in: *Proc. 47th European Solid-State Device Research Conference (ESSDERC),* 2017.

[76] Mahmoodi, M. R., *et al.* RX-PUF: Low Power, Dense, Reliable, and Resilient Physically Unclonable Functions Based on Analog Passive RRAM Crossbar Arrays. in: *Proc. Symposium on VLSI Technology*, 2018.

[77] Gao, L. *et al.* Physical unclonable function exploiting sneak paths in resistive cross-point array. *IEEE Transactions on Electron Devices* **63**, 2016.

[78] Chakrabartty, S. and Cauwenberghs, G. Sub-microwatt analog VLSI trainable pattern classifier. *IEEE J.Solid-State Circuits* **42**, 2007.

[79] Ramakrishnan, S. and Hasler, J. Vector-matrix multiply and winner-take-all as an analog classifier. *IEEE Trans. Very Large Scale Integr. Syst.* **22**, 2014.

[80] Eryilmaz, S. B., *et al.* Brain-like associative learning using a nanoscale non-volatile phase change synaptic device array. *Front. Neurosci*. **8**, 2014.

[81] Burr, G. W., *et al.* Experimental demonstration and tolerancing of a large-scale neural network (165 000 synapses) using phase-change memory as the synaptic weight element. *IEEE Trans. Electron Devices* **62**, 2015.

[82] Boybat, I., *et al.* Neuromorphic computing with multi-memristive synapses. *Nature Communications* **9**, 2018.

[83] Kaneko, Y., Nishitani, Y. and Ueda, M. Ferroelectric artificial synapses for recognition of a multishaded image. *IEEE Trans. Electron Devices* **61**, 2014.

[84] Boyn, S., *et al.* Learning through ferroelectric domain dynamics in solid-state synapses. *Nature Communications* **8**, 2017.

[85] Romera, M., *et al.* Vowel recognition with four coupled spin-torque nano-oscillators. *Nature* **563**, 2018.

[86] Milano, G., *et al.* Self-limited single nanowire systems combining all-in-one memristive and neuromorphic functionalities. *Nature Communications* **9**, 2018.

[87] Choi, S., *et al.* SiGe epitaxial memory for neuromorphic computing with reproducible high performance based on engineered dislocations. *Nature Materials* **17**, 2018.

[88] Kim, K. H., *et al.* A functional hybrid memristor crossbar-array/CMOS system for data storage and neuromorphic applications. *Nano Letters* **12**, 2012.

[89] Yeon, H., *et al.* Alloying conducting channels for reliable neuromorphic computing. *Nature Nanotechnology* **15**, 2020.

[90] Fuller, E. J., *et al.* Parallel programming of an ionic floating-gate memory array for scalable neuromorphic computing. *Science* **364**, 2019.

[91] Goswami, S., *et al.* Robust resistive memory devices using solution-processable metal-coordinated azo aromatics. *Nat. Mater.* **16**, 2017.

[92] Indiveri, G., *et al.* Integration of nanoscale memristor synapses in neuromorphic computing architectures. *Nanotechnology* **24**, 2013.

[93] Ambrogio, S., *et al.* Neuromorphic learning and recognition with one-transistor-one-resistor synapses and bistable metal oxide RRAM. *IEEE Trans. Electron Devices* **63**, 2016.

[94] NVidia corporation investor day presentation (2017).

[95] ARM ML processor (https://www.arm.com/products/processors/ machine-learning); Intel Mobileye (https://www.mobileye.com/en-us/); Google Edge TPU (https://cloud.google.com/edge-tpu/).

[96] Moons, B., *et al*. Envision: A 0.26-to-10 TOps/W subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm FDSOI. *ISSCC*, 2017.

[97] Marinella, M. J., *et al.* Multiscale co-design analysis of energy, latency, area, and accuracy of a ReRAM analog neural training accelerator. *JETCAS* **8**, 2018.

[98] Li, C., *et al.* Analogue signal and image processing with large memristor crossbars. *Nat. Electron.* **1**, 2018.

[99] Hasler, P., *et al.* Single transistor learning synapses. *Adv. Neural Inf. Process. Syst.* 1994.

[100] Lu. J., *et al.* A 1 TOps/W analog deep machine-learning engine with floating-gate storage in 0.13 μm CMOS. *IEEE J. Solid-State Circuits* **50**, 2015.

[101] Merrikh-Bayat, F., *et al.* High-performance mixed-signal neurocomputing with nanoscale floating-gate memory cell arrays. *IEEE transactions on neural networks and learning systems* **29**, 2017.

[102] Alibart, F., *et al.* High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm, Nanotechnology **23**, 2012.

[103] Hu, M., *et al.* Memristor-based analog computation and neural network classification with a dot product engine. *Adv. Mater.* **30**, 2018.

[104] Xia, Q., and Yang, J. J. Memristive crossbar arrays for brain-inspired computing. *Nat. Mater.* **18**, 2019.

[105] Sheridan, P. M., *et al.* Sparse coding with memristor networks. *Nat. Nanotechnol*. **12**, 2017.

[106] Cai, F., *et al.* A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations. *Nat. Electron.* **2**, 2019.

[107] Biju, K. P., *et al.* Resistive switching characteristics and mechanism of thermally grown WO_x thin films. *J. Appl. Phys.* **110**, 2011.

[108] Kim, K., *et al.* A functional hybrid memristor crossbar-array/CMOS system for data storage and neuromorphic applications. *Nano Lett.* **12**, 2011.

[109] Choi, S., *et al.* Experimental demonstration of feature extraction and dimensionality reduction using memristor networks. *Nano Lett.* **17**, 2017.

[110] Jeong Y., *et al.* K-means data clustering with memristor networks. *Nano Lett.* **18**, 2018.

[111] Liu, Q., *et al.* A fully integrated analog ReRAM based 78.4 TOps/W compute-in-memory chip with fully parallel MAC computing. in: *Proc. IEEE International Solid-State Circuits Conference (ISSCC)*, 2020.

[112] Zheng, X., *et al.* Error-resilient analog image storage and compression with analog-valued RRAM arrays: An adaptive joint source-channel coding approach. in: *Proc. IEEE International Electron Device Meeting (IEDM),* 2018.

[113] Kim, H., *et al.* 4K-memristor analog-grade passive crossbar circuit. *arXiv preprint arXiv:1906.12045*, 2019.

[114] Alibart, F., *et al.* High-precision tuning of state for memristive devices by adaptable variation-tolerant algorithm. *Nanotechnology* **23**, 2012.

[115] Nili, H., *et al.* Comprehensive compact phenomenological modeling of integrated metal-oxide memristors. *IEEE Transactions on Nanotechnology* **19**, 2020.

[116] Ielmini, D. and R. Waser, Resistive switching: from fundamentals of nanoionic redox processes to memristive device applications. *John Wiley & Sons*, 2015.

[117] Mahmoodi, M. R., *et al.* The Impact of Device Uniformity on Functionality of Analog Passively-Integrated Memristive Circuits. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2021. (under review)

[118] Klachko, M., *et al.* Improving noise tolerance of mixed-signal neural networks. in: *Proc. of International Joint Conference on Neural Networks (IJCNN)*, 2019.

[119] Young-Fisher, K. G., *et al.* Leakage current-forming voltage relation and oxygen gettering in HfO x RRAM devices. *IEEE electron device letters* **34**, 2013.

[120] Fahimi, Z., *et al.* Mitigating Imperfections in Mixed-Signal Neuromorphic Circuits. *Nature Communications*, 2021.

[121] Agarwal, S., *et al.*, Compensating for parasitic voltage drops in resistive memory arrays. in: *Proc. IMW'17*, 2017.

[122] Kim, T., *et al*. Input voltage mapping optimized for resistive memory-based deep neural network hardware. *IEEE Electron Device Letters* **38**, 2017.

[123] Jeong, Y., *et al*. Parasitic effect analysis in memristor-array-based neuromorphic systems, *IEEE Trans. Nanotechnol.* **17**, 2018.

[124] Xia, L., *et al.* Technological exploration of RRAM crossbar array for matrix-vector multiplication. *J. Comp. Sci. Technol*. **31**, 2016.

[125] Hu, M., *et al.* Dot-product engine for neuromorphic computing: Programming 1T1M crossbar to accelerate matrix-vector multiplication. in: *Proc. DAC'16*, 2016.

[126] Yan, B., *et al.* Understanding the trade-offs of device, circuit and application in ReRAM-based neuromorphic computing systems. in: *Proc. IEDM'17*, 2017.

[127] Merrikh Bayat, F., *et al.* Memristor-based perceptron classifier: Increasing complexity and coping with imperfect hardware. in: *Proc. ICCAD'17*, 2017.

[128] Chen, A. A comprehensive crossbar array model with solutions for line resistance and nonlinear device characteristics. *IEEE Trans. Electron Devices* **60**, 2013.

[129] Bavandpour, M., *et al.* Mixed-signal neuromorphic processors: Quo vadis?. in: *Proc. IEEE S3S'19*, Dec. 2019.

[130] Mahmoodi, M. R., and Dmitri Strukov. An ultra-low energy internally analog, externally digital vector-matrix multiplier based on NOR flash memory technology. in: *Proc. of the 55th Annual Design Automation Conference*. 2018.

[131] Sedra, A. S. and K. C. Smith. A second-generation current conveyor and its applications. *IEEE Trans. Circ. Theory* **17**, 1970.

[132] Salama, K. N. and A. M. Soliman. CMOS operational transresistance amplifier for analog signal processing. *Microelectronics Journal* **30**, 1990.

[133] Brunn, E. CMOS high speed, high precision current conveyor and current feedback amplifier structures. *Int. Journal of Electronics* **74**, 1993.

[134] Calvo, B., *et al.* High-speed high-precision CMOS current conveyor. *Analog Integrated Circuits and Signal Processing* **34**, 2003.

[135] D'Angelo R. and S. Sonkusale. A time-mode translinear principle for nonlinear analog computation. *IEEE Transactions on Circuits and Systems I: Regular Papers* **62**, 2015.

[136] Bankman D. and B. Murmann. An 8-bit, 16 input, 3.2 pJ/Op switched-capacitor dot product circuit in 28-nm FDSOI CMOS. in: *proc. Solid-State Circuits Conference* (A-SSCC), 2016.

[137] Lee E. and S. Wong. Analysis and design of a passive switched-capacitor matrix multiplier for approximate computing. *IEEE Journal of Solid-State Circuits* **52**, 2017.

[138] Binas, J., *et al.* Precise deep neural network computation on imprecise low-power analog hardware. *arXiv:1606.07786,* 2016.

[139] Kim, M. and P. Hanumolu. A 10 MS/s 11-bit 0.19 mm^2 algorithmic ADC with improved clocking scheme. *IEEE Journal of Solid-State Circuits* **44**, 2009.

[140] Razavi, B. The StrongARM latch [a circuit for all seasons]. IEEE Solid-State Circuits Magazine **7**, 2015.

[141] Lee, E. and S. Wong. Analysis and design of a passive switched-capacitor matrix multiplier for approximate computing. *IEEE Journal of Solid-State Circuits* **52**, 2017.

[142] Bavandpour, M., M. R. Mahmoodi, and D. B. Strukov. aCortex: An Energy-Efficient Multipurpose Mixed-Signal Inference Accelerator. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits* **6**, 2020.

[143] Bavandpour, M., *et al.* Mixed-signal vector-by-matrix multiplier circuits based on 3d-nand memories for neurocomputing. in: *proc. 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2020.

[144] Sahay, S., *et al.* A 2T-1R Cell Array with High Dynamic Range for Mismatch-Robust and Efficient Neurocomputing. in: *proc. 2020 IEEE International Memory Workshop (IMW)*, 2020.

[145] Faisal, A., L. P. Selen, and D. M. Wolpert. Noise in the nervous system. *Nature Review Neuroscience* **9**, 2008.

[146] Rolls, E. and G. Deco. The Noisy Brain: Stochastic Dynamics as a Principle of a Brain Function, Oxford University Press, 1st Edition, 2010.

[147] White, J., J. T. Rubinstein, and A.R. Kay. Channel noise in neurons. *Trends in Neurosciences*. **23**, 2000.

[148] Branco, T. and K. Staras. The probability of neurotransmitter release: Variability and feedback control at single synapses. *Nature Review Neuroscience* **10**, 2009.

[149] Stein, R., E.R. Gossen, and K.E. Jones. Neuronal variability: Noise or part of the signal?, *Nature Review Neuroscience* **6**, 2005.

[150] Yarom Y. and J. Hounsgaard. Voltage fluctuations in neurons: Signal or noise. *Physiological Reviews* **91**, 2011.

[151] Czanner, G., S.V. Sarma, D. Ba, U.T. Eden, W. Wu, E. Eskandar, H. H. Lim, S. Temereanca, W. A. Suzuki, and E.N. Brown. Measuring the signal-to-noise ratio of a neuron. *Proceedings of National Academy of Sciences* **112**, 2015.

[152] Levy, W., and R.A Baxter. Energy efficient neuronal computation via quantal synaptic failures. *Journal of Neuroscience* **22**, 2002.

[153] Smolensky, P. Information processing in dynamical systems: Foundations of harmony theory, in: Rumelhart, David E.; McLelland, James L. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* **1**, 1986.

[154] Hinton, G. and T. J. Sejnowski, Optimal perceptual inference. in: *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR'83)*, 1983.

[155] Hopfield, J. Neural networks and physical systems with emergent collective computational abilities. in: *Proc. National Academy of Sciences of the USA* **79**, 1982.

[156] Smith, K. Neural networks for combinatorial optimization: A review of more than a decade of research. *Journal on Computing* **11**, 1999.

[157] Fischer, A., and C. Igel. An introduction to restricted Boltzmann machines. in: *Proc. Iberoamerican Congress on Pattern Recognition (CIARP'12)*, 2012.

[158] Hinton, G. A practical guide to training restricted Boltzmann machines. in: G. Montavon, G. B. Orr, and K.-R. Muller. *Neural networks: Tricks of the trade* **7700***,* 2012.

[159] Salakhutdinov, R., A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. in: *Proc. International Conference on Machine Learning (ICML'07)*, 2007.

[160] Salakhutdinov, R., A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. in: *Proc. International Conference on Machine Learning (ICML'07)*, 2007.

[161] Gao, L., *et al.* Digital-to-analog and analog-to-digital conversion with metal oxide memristors for ultra-low power computing. in: *Proc. IEEE International Symposium on Nanoscale Architectures (NanoArch'13)*, 2013.

[162] Mahmoodi, M. R., *et al.* Intrinsic bounds for computing precision in memristor-based vector-by-matrix multipliers. *IEEE Transactions on Nanotechnology* 19 (2020): 429-435.

[163] Mahmoodi, M. R., and D. Strukov. Breaking POps/J barrier with analog multiplier circuits based on nonvolatile memories. in *Proceedings of the International Symposium on Low Power Electronics and Design*, 2018.

[164] Bavandpour, M., M. R. Mahmoodi, and D. B. Strukov. Energy-efficient time-domain vector-by-matrix multiplier for neurocomputing and beyond. *IEEE Transactions on Circuits and Systems II: Express Briefs* **66**, 2019.

[165] Fukami, S., and H. Ohno, Perspective: Spintronic synapse for artificial neural network. *Journal of Applied Physics* **124**, 2018.

[166] Debashis, P., *et al.* Experimental demonstration of nanomagnet networks as hardware for Ising computing. in: *Proc. IEEE International Electron Devices Meeting (IEDM'17)*, 2017.

[167] Serb, A., J. Bill, A. Khiat, R. Berdan, R. Legenstein, and T. Prodromakis. Unsupervised learning in probabilistic neural networks with multi-state metal-oxide memristive synapses. *Nature Communications* **7**, 2016.

[168] Babu, A., S. Lashkare, U. Ganguly and B. Rajendran. Stochastic learning in deep neural networks based on nanoscale PCMO device characteristics. *Neurocomputing* **321**, 2018.

[169] Ielmini, D., Resistive switching memories based on metal oxides: mechanisms, reliability and scaling. *Semiconductor Science and Technology* **31**, 2016.

[170] Hung, K., P.K. Ko, C. Hu, and Y.C. Cheng. A physics-based MOSFET noise model for circuit simulators. *IEEE Trans. on Electron Devices* **37**, 1990.

[171] Li, Z., J. Ma, Y. Ye, and M. Yu. Compact channel noise models for deep-submicron MOSFETs. *IEEE Trans. on Electron Devices* **56**, 2009.

[172] Chou, C., Z.-J. Lin, P.-L. Tseng, C.-F. Li, C.-Y. Chang, W.-C. Chen, Y.-D. Chih, and T.-Y.J. Chang. An N40 256K × 44 embedded RRAM macro with SL-precharge SA and low-voltage current limiter to improve read and write performance. in: *Proc. International Solid-State Circuits Conference (ISSCC'18)*, 2018.

[173] Ramanujam, J., and P. Sadayappan. Mapping combinatorial optimization problems onto neural networks. *Information Science* **82**, 1995.

[174] King, A., *et al.* Observation of topological phenomena in a programmable lattice of 1,800 qubits. *Nature* **560**, 2018.

[175] Tsymbal, E.Y., A. Gruverman, V. Garcia, M. Bibes, and A. Barthélémy. Ferroelectric and multiferroic tunnel junctions, *MRS Bulletin* **37**, 2012.

[176] Elidan, G., M. Ninio, Nir Friedman, and Dale Schuurmans. Data perturbation for escaping local maxima in learning, *AAAI/IAAI*, 2002.

[177] Ninio, M., and J. J. Schneider. Weight annealing. *Physica A: Statistical Mechanics and its Applications* **349**, 2005.

[178] Loh, K., B. Golden, and E. Wasil. Solving the one-dimensional bin packing problem with a weight annealing heuristic. *Computers and Operations Research* **35**, 2008.

[179] Loh, K., Bruce Golden, and Edward Wasil. A Weight Annealing Algorithm for Solving Two-dimensional Bin Packing Problems. *Operations Research and Cyber-Infrastructure*, 2009.

[180] Charon, I., and O. Hudry. The noising method: A new method for combinatorial optimization. *Operations Research Letters*, 1993.

[181] Coy, S. P., B.L. Golden, and E.A. Wasil. A computational study of smoothing heuristics for the traveling salesman problem. *European Journal of Operations Research* **124**, 2000.

[182] Coy, S. P., B. L. Golden, G. C.Runger, and E. A. Wasil. Using experimental design to find effective parameter settings for heuristics. *Journal of Heuristics* **7**, 2001

[183] Ushijima-Mwesigwa, H., C. Negre, and S. M. Mniszewski. Graph partitioning using quantum annealing on the d-wave system. in: *Proc. of the Second International Workshop on Post Moores Era Supercomputing*, 2017.

[184] Mahmoodi, M. R., Z. Fahimi, H. Nili, V. Polishchuk, D. B. Strukov. Combinatorial Optimization by Weight Annealing in Memristive Hopfield Networks. *Nature Scientific Reports*, 2021.

[185] Assaad, R. S., and J. Silva-Martinez. The recycling folded cascode: A general enhancement of the folded cascode amplifier, *IEEE Journal of Solid-State Circuits* **44**, 2009.

[186] Marandi, A., *et al.* Network of time-multiplexed optical parametric oscillators as a coherent Ising machine. *Nature Photonics* **8**, 2014.

[187] Wang, Z., *et al.* Coherent Ising machine based on degenerate optical parametric oscillators. *Physical Review A* **88**, 2013.

[188] McMahon, P., *et al.* A fully programmable 100-spin coherent Ising machine with all-to-all connections. *Science* **354**, 2016.

[189] Razavi, B. The flash adc [a circuit for all seasons]. *IEEE Solid-State Circuits Magazine* **9**, 2017.

[190] Lazzaro, J., *et al.* Winner-take-all networks of O (n) complexity. 1988.

[191] Razavi, B. 12.4 PTAT current generation, *Design of Analog CMOS Integrated Circuits*, 1st ed., 2001.

[192] Wu, C., *et al.* A low TC, supply independent and process compensated current reference. in *Proc. Custom Integrated Circuits Conference*, 2015.

[193] Osipov, D., and S. Paul. Temperature-Compensated β -Multiplier Current Reference Circuit. in *IEEE Transactions on Circuits and Systems II: Express Briefs* **64**, 2017.

[194] Lee, J., and S. Cho. A 1.4-µW 24.9-ppm/°C Current Reference With Process-Insensitive Temperature Compensation in 0.18-µm CMOS. in *IEEE Journal of Solid-State Circuits* **47**, 2012.

[195] Serrano, G., and P. Hasler. A Precision Low-TC Wide-Range CMOS Current Reference. in *IEEE Journal of Solid-State Circuits* **43**, 2008.

[196] Choi, M., *et al.* A 23pW, 780ppm/°C resistor-less current reference using subthreshold MOSFETs. in *Proc. European Solid State Circuits Conference*, 2014.

[197] Wang, H., and P. P. Mercier. A 3.4-pW 0.4-V 469.3 ppm/°C Five-Transistor Current Reference Generator. *IEEE Solid-State Circuits Letters* **1**, 2018.

[198] Dong, Q., *et al.* A 1.02nW PMOS-only, trim-free current reference with 282ppm/°C from −40°C to 120°C and 1.6% within-wafer inaccuracy. in: *Proc. IEEE European Solid State Circuits Conference*, 2017.

[199] Arora, N. D., J. R. Hauser and D. J. Roulston. Electron and hole mobilities in silicon as a function of concentration and temperature. *IEEE Transactions on Electron Devices* **29**, 1982.

[200] Bendali, A., and Y. Audet. A 1-V CMOS Current Reference With Temperature and Process Compensation. *IEEE Transactions on Circuits and Systems I: Regular Papers* **54**, 2007.

[201] Chen, H., and A. F. Murray. Continuous restricted Boltzmann machine with an implementable training algorithm. *IEE Proc.-Vis. Image Signal Process*. **150**, 2003.

[202] Gubbi, J., R. Buyya, S. Marusic, M. Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future generation computer systems* **29**, 2013.

[203] Rührmair, U., F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber Modeling attacks on physical unclonable functions. in *Proc. 17th ACM conference on Computer and communications security,* 2010.

[204] Maes, R., and I. Verbauwhede, Physically unclonable functions: A study on the state of the art and future research directions. *Towards Hardware-Intrinsic Security,* 2010.

[205] Rührmair, U., H. Busch, and S. Katzenbeisser. Strong PUFs: models, constructions, and security proofs. *Towards hardware-intrinsic security*. 2010.

[206] Lim, D., J. W. Lee, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, Extracting Secret Keys from Integrated Circuits. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* **13**, 2005.

[207] Rührmair, U., J. Sölter, and F. Sehnke. On the Foundations of Physical Unclonable Functions, *Technical Report* **227***, IACR Cryptology E-print Archive,* 2009.

[208] Maes, R. Physically unclonable functions: Constructions, properties, and applications. *Katholieke Universiteit Leuven*, 2012.

[209] Handschuh, H., G. Schrijen, and P. Tuyls. Hardware intrinsic security from physically unclonable functions. *Towards Hardware-Intrinsic Security*, 2010.

[210] Rostami, M., J. B. Wendt, M. Potkonjak and F. Koushanfar. Quo vadis, PUF?: trends and challenges of emerging physical-disorder based security. in *Proc. Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2014.

[211] Mahmoodi, M. R., *et al.* Ultra-low power physical unclonable function with nonlinear fixed-resistance crossbar circuits. in *Proc. IEEE International Electron Devices Meeting (IEDM)*, 2019.

[212] Jeloka, S., *et al.* A sequence dependent challenge-response PUF using 28nm SRAM 6T bit cell. in *Proc. Symp. on VLSI Circuits*, 2017.

[213] Alvarez, A., W. Zhao and M. Alioto. 14.3 15fJ/b static physically unclonable functions for secure chip identification with <2% native bit instability and 140× Inter/Intra PUF hamming distance separation in 65nm. in *Proc. IEEE International Solid-State Circuits Conference-(ISSCC)*, 2015.

[214] Mahmoodi, M. R., *et al*. A Strong Physically Unclonable Function With> 2^{80} CRPs and< 1.4% BER Using Passive ReRAM Technology. *IEEE Solid-State Circuits Letters* **3,** 2020.

[215] Gao, Y. *et al.* Lightweight (reverse) fuzzy extractor with multiple reference PUF responses, *IEEE Trans. Inf. Forensics Secur.* **14**, 2018.

[216] Xi, X., H. Zhuang, N. Sun, and M. Orshansky. Strong subthreshold current array PUF with 265 challenge-response pairs resilient to machine learning attacks in 130nm CMOS. in *Proc. Symp. on VLSI Circuits*, 2017.

[217] Mathew, S. K., *et al.* μRNG: A 300–950 mV, 323 Gbps/W all-digital full-entropy true random number generator in 14 nm FinFET CMOS. *IEEE J. Solid-State Circuits* **51**, 2016.

[218] Che, W., *et al.* Novel offset techniques for improving bitstring quality of a hardware-embedded delay PUF. *IEEE Trans. Very Large Scale Integr. (VLSI)* **26**, 2018.

[219] Mahmoodi, M. R., *et al.* ChipSecure: A reconfigurable analog eFlash-based PUF with machine learning attack resiliency in 55nm CMOS. in *Proc*. 56th *ACM/IEEE Design Automation Conference (DAC),* 2019.

[220] Larimian, S., M. R. Mahmoodi, and D. B. Strukov. Lightweight Integrated Design of PUF and TRNG Security Primitives Based on eFlash Memory in 55-nm CMOS. *IEEE Transactions on Electron Devices* **67**, 2020.