APL Materials

$Pr_xCa_{1-x}MnO_3$ based stochastic neuron for Boltzmann machine to solve "maximum cut" problem

Cite as: APL Mater. **7**, 091112 (2019); https://doi.org/10.1063/1.5108694
Submitted: 01 May 2019 • Accepted: 05 September 2019 • Published Online: 24 September 2019

Devesh Khilwani, Vineet Moghe, Sandip Lashkare, et al.

$Pr_xCa_{1-x}MnO_3$ based stochastic neuron for Boltzmann machine to solve "maximum cut" problem

Cite as: APL Mater. 7, 091112 (2019); doi: 10.1063/1.5108694
Submitted: 1 May 2019 • Accepted: 5 September 2019 •
Published Online: 24 September 2019

Devesh Khilwani,[1,a)] Vineet Moghe,[1,a)] Sandip Lashkare,[1] Vivek Saraswat,[1] Pankaj Kumbhare,[1] Maryam Shojaei Baghini,[1] Srivatsava Jandhyala,[2] Sreenivas Subramoney,[2] and Udayan Ganguly[1]

AFFILIATIONS

[1]Department of Electrical Engineering, IIT Bombay, Mumbai, India
[2]Processor Architecture Research Lab, Intel Labs, Bangalore, India

Note: This paper is part of the Special Topic on Emerging Materials in Neuromorphic Computing.
[a)]**Contributions:** D. Khilwani and V. Moghe contributed equally to this work.

ABSTRACT

The neural network enables efficient solutions for Nondeterministic Polynomial-time (NP) hard problems, which are challenging for conventional von Neumann computing. The hardware implementation, i.e., neuromorphic computing, aspires to enhance this efficiency by custom hardware. Particularly, NP hard graphical constraint optimization problems are solved by a network of stochastic binary neurons to form a Boltzmann Machine (BM). The implementation of stochastic neurons in hardware is a major challenge. In this work, we demonstrate that the high to low resistance switching (*set*) process of a $Pr_xCa_{1-x}MnO_3$ (PCMO) based RRAM (Resistive Random Access Memory) is probabilistic. Additionally, the voltage-dependent probability distribution approximates a sigmoid function with 1.35%–3.5% error. Such a sigmoid function is required for a BM. Thus, the Analog Approximate Sigmoid (AAS) stochastic neuron is proposed to solve the maximum cut—an NP hard problem. It is compared with Digital Precision-controlled Sigmoid (DPS) implementation using (a) pure CMOS design and (b) hybrid (RRAM integrated with CMOS). The AAS design solves the problem with 98% accuracy, which is comparable with the DPS design but with 10× area and 4× energy advantage. Thus, ASIC neuro-processors based on novel analog neuromorphic devices based BM are promising for efficiently solving large scale NP hard optimization problems.

I. INTRODUCTION

The conventional von Neumann computer based on deterministic CMOS logic implementation has been extremely successful in implementing sequential algorithms using clearly demarcated processing and memory units.[1,2] However, there are many important problems such as graphical constraint optimization, factorization, and other Nondeterministic Polynomial-time (NP)-hard problems which do not have a polynomial time algorithm to find globally optimal solutions. A serial search through a large number of states leads to a memory and computing resource challenge. Hence, there has been growing interest to test alternative computing paradigms.[3,4] Brain inspired artificial neural networks have shown immense promise for efficiently searching approximate solutions and have found wide applications in pattern recognition and optimization problems.[5,6] Specifically, the Hopfield-Tank networks allow a parallel scan of possible network states and have been algorithmically shown to estimate solutions for the classical Traveling Salesman Problem (TSP).[7]

More recently, Spiking Neural Networks (SNNs) were proposed as a biologically more accurate model, which use spikes in neurons as information carriers along with plastic connections called synapses.[8] Apart from parallel communication through spikes, other features of the SNNs include stochastic spiking and[9,10] refractory periods.[11] These three biological features are useful in various ways, e.g., constraint optimization,[11] enhanced learning,[9] and sensing.[10] Measured data from isolated retina of larval tiger salamander shows that the refractory period regulates the spike rate

of the ganglion cells located near the inner surface of the retina.[12] The refractory period of a neuron can also help escape local minima while performing stochastic optimization tasks using neural networks.[11]

Parallel computation with realtime information exchange between neurons produces a communication bottleneck in von Neumann computers with separate logic and memory blocks connected with a bus. To fully realize the potential of these models, dedicated hardware and architecture have been proposed. An excellent review of neuromorphic algorithms and hardware for constraint satisfaction problems has been presented.[13] Nanoscale devices have been used to further enhance the efficiency of these solutions. For example, coupled oscillators[14–16] and memristor crossbar array based Boltzmann machines (BMs)[17] have been explored.

The Boltzmann machine (BM) is an important class of neural network, where neurons are typically binary (i.e., spike is "1" and no spike is "0"). The n neurons in the network can choose the ith state out of 2^n states with a probability (p_i) that depends on the energy (E_i) of the state based on the Boltzmann distribution, i.e., $p_i \propto \exp(-E_i)$. Constraint optimization problems are solved by mapping the constraints to network architecture and the energy function (E_i).[18] Furthermore, neurons in the BM can be modeled as a Markov chain to solve the traveling salesman problem[11,19] of the NP hard class of problems. The key to realizing a Markov chain based model lies in being able to generate random numbers according to a given function. A sigmoid function for the stochastic neuron model was proposed earlier.[19]

Neuromorphic engineering aspires to implement such promising algorithms in hardware to enable performance, power, and area efficiency advantages. To enable BM in hardware, the challenge is to implement a stochastic neuron. On the other hand, analog synapses have been explored in detail as shown in the literature review.[20] Various neuron designs have been explored in the literature.[21] Circuits used to implement silicon neurons have been reviewed.[22] An optimal mix of analog and digital design is required to achieve brainlike efficiency in computing.[23] A low power analog LIF (Leaky Integrate and Fire) neuron using novel physics in traditional silicon-on-insulator Metal Oxide Semiconductor Field Effect Transistor (MOSFET) has been demonstrated[24] but provides rather miniscule stochasticity.[25]

The nanoscale devices like memristors show enhanced stochastic switching[26–28] without requiring circuit based amplification of noise.[29] Furthermore, analog matrix multiplication based on the memristor crossbar has been shown as significantly superior to the digital version for the Boltzmann machine.[17] Various nanoscale device based stochastic neurons have been demonstrated. A combined neuro-synaptic core was proposed using a memristive magnetic tunnel junction device.[30] The magnetization switching driven by spin-transfer torque in combination with back-hopping was used to demonstrate stochastic current spike generation. However, the switching resistance ratio was poor and stochasticity (without an external magnetic field support) was experimentally observed for only low temperatures (T ~ 130 K) and very high current densities [100× > than PCMO RRAM (Resistive Random Access Memory)]. Another general-purpose weight storage element and stochastic neuron model was proposed using a TiO_2 memristor.[31] The resistive switching, similar to PCMO, was achieved through vacancy modulation; however, this device required electroforming step for operation and higher operating voltages, both of which negatively impact device variability and endurance. Detection of temporal correlations in parallel data streams was proposed using a stochastic phase change neuron in $Ge_2Sb_2Te_5$ Phase Change Memory (PCM).[32] The device resistance was a function of the crystalline vs amorphous phase thicknesses of the PCM layer and the uncertainty in melt-quench amorphization reset step was the primary source of stochasticity in neuron operation. However, the input to this neuron needed to be converted to a series of crystallization pulses (as opposed to a single fixed pulse for PCMO) which requires extra peripheral circuitry and renders low feasibility to network level integration. More recently, low barrier magnets have been proposed for stochastic switching in a 1T/1M arrangement.[33–35] However, these devices have very stringent fabrication constraints of near-critical thickness magnetization layer or circular magnets for an absence of preferential magnetic orientation which are a challenge for nanoscale production and often require noise amplification inverters at the output. Yet another implementation using an electroforming free VO_2 Mott memristor based stochastic neuron was demonstrated recently,[36] but it required additional noise to the input for exhibiting stochastic behavior. Fidelity to neuronal models like stochastic Hodgkin Huxley[37] and spike response models[38] has also been demonstrated. With respect to applications, stochastic neurons have been used for enhanced sensing,[32] training, and recognition of the logic function[39,40] and on datasets such as MNIST, CIFAR, etc.,[41–44] with promising energy benchmarks;[45] however, the application to NP hard constraint optimization has not been explored.

Unlike stochasticity in filamentary RRAMs[26] which produce binary states,[46] PCMO ($Pr_xCa_{1-x}MnO_3$) is a nonfilamentary RRAM to enable analog memory with forming-less operation and area scalable currents with good endurance and retention.[47,48] Excellent low energy, analog PCMO synapses have been demonstrated.[49–52] Integrate-and-fire (IF) neurons have been demonstrated based on the *set* (high to low resistance switching) process.[53] Thus, PCMO RRAM provides a materials system, which provides both analog synaptic and IF neuronal functionality. However, the stochastic switching of PCMO based RRAM and its utilization in stochastic neurons have not been presented earlier.

In this work, we present a stochastic neuron based on PCMO RRAM for a BM to solve an NP hard problem, i.e., Maximum Cut (or Max-Cut). First, we experimentally show that PCMO RRAM has approximately sigmoid switching probability with voltage. We utilize the natural analog approximation of sigmoid stochasticity to design a compact neuron. A comparison with digital precision-controlled sigmoid stochasticity is presented with purely CMOS as well as CMOS with integrated RRAM implementations. We have considered 65 nm CMOS technology for the current document; however, the analysis is fairly general. We show that the networks sample from the Boltzmann distribution approximately. We compare the performance in terms of accuracy of solution of Max-Cut. Finally, we present the area and power benefits.

II. BOLTZMANN MACHINE ALGORITHM

A BM is a fully connected network of n binary neurons [Fig. 1(c)] described in the literature.[11] A weight is associated with each connection and a bias associated with each neuron. The state

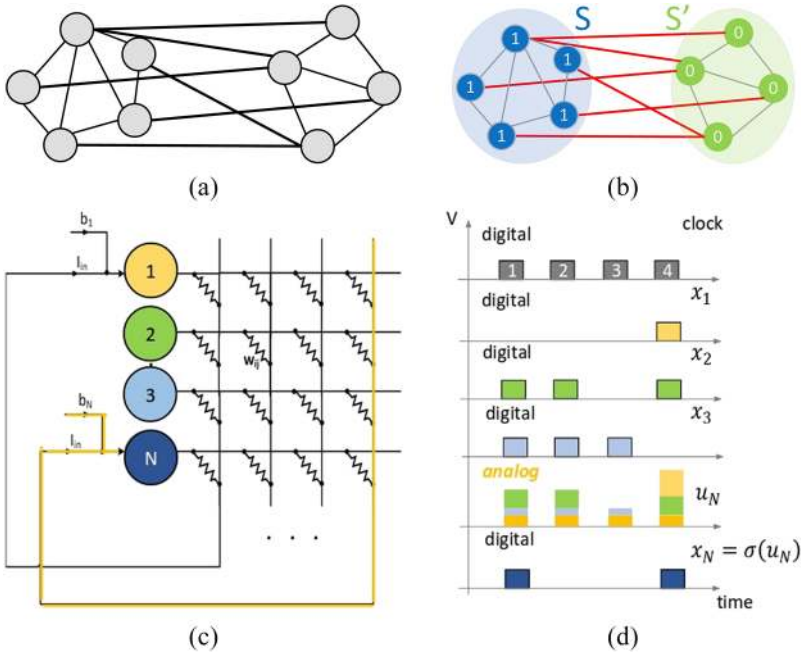

FIG. 1. (a) An instance of Max-Cut where the edge weight is proportional to the edge length. (b) The solution is a partitioned graph in two sets (S, S′) so that the sum of edge weights between the sets (red edges) is maximized. (c) A crossbar implementation of the network used to solve the Max-Cut problem, while in (d) the signal processing is described. In the crossbar, neurons (1–3) may issue digital spikes at digital clock times. These digital spikes produce current through analog weights (W_{ij}) which are summed along with bias. This analog current is converted into an analog voltage $u_N = \sum_{i=1}^{3} W_{iN}x_i + b_N$ as an input to neuron N, which fires digital spikes stochastically $x_N = \sigma(u_N)$. For example, although the input u_N is identical at time steps 1 and 2, the output of neuron N, i.e., x_N does not produce identical value. (a) Max-Cut graph and (b) Max-Cut partition.

of the network can be expressed as a binary vector which represents neuron in binary states, i.e., on ("1") or off ("0"). Such a state $\vec{x}$ (among 2^n possible states) occurs with a probability given by

$$P(\vec{x}) = \frac{1}{Z}\exp(-E_{\vec{x}}/kT), \tag{1}$$

where

$$-E_{\vec{x}}/kT = \sum_{i=1}^{N} b_i * x_i + \sum_{i=1}^{N}\sum_{j=i+1}^{N} w_{ij} * x_i * x_j. \tag{2}$$

Here, Z is the normalization factor, b_i is the bias of the ith neuron, x_i is the state of the ith neuron (0/1), and w_{ij} is the weight of the connection between ith and jth neurons. Thus, the input (u_i) to the ith neuron is the change in network energy caused by its switching, which is given by the following:

$$u_i = \Delta E_i = b_i + \sum_j w_{ij} * x_j. \tag{3}$$

The crossbar array shown in Fig. 1(c) functions to compute this sum and feed it to the neuron. The weights are summed along a column by Kirchhoff's law of networks as input to the neuron, along with a self-bias current. The clocked implementation is shown in Fig. 1(d) where the input from digital neurons is converted to analog current through memristors, summed through the crossbar to generate an analog u_i, which is used by the stochastic neuron to issue digital spikes.[17] Equation (1) indicates that the BM will visit the lowest energy state the most frequently. Thus, we need to map the cost of an optimization problem to the energy of the network so that the most frequently visited state is the minimum cost solution. The daunting class of NP-hard problems is challenging to solve in the serially processed von Neumann computing approach. However, BM in hardware provides a way to exchange information between all neurons in parallel. The parallel information processing in BMs may provide more efficient solutions.

A. Markov chain model for stochastic neuron

Neuron's operation may be represented as a Markov chain,[11] as shown in Fig. 2(a). In state 0, the output is 0. Here, the neuron accepts input u to *stochastically* decide whether to transition to the state τ or stay in the same state. The transition to state τ occurs with sigmoid probability dependence on input u, i.e., $\sigma(u)$. Hence, the neuron stays in the same state with probability $1 - \sigma(u)$. Once a neuron has reached the state τ, it *deterministically* transitions to the next states until it reaches state 1. In all these states from state τ to state 1, the output remains 1. On reaching state 1, the neuron again *stochastically* decides to transition to state τ [with probability $\sigma(u)$] or state 0 [with probability $1 - \sigma(u)$] depending on the input u. Given such a neuron, the network visits states such that it samples from a Boltzmann distribution.[19]

B. Network definition for the Max-Cut problem

Next, we describe the solution of Max-Cut, which is one of the first problems to be demonstrated as NP hard and has many practical applications, e.g., resource maximization in networks.[54] In solving the weighted Max-Cut problem on a graph, the aim is to cut the graph in two parts such that the sum of edge weights crossing between the two parts is maximized. To do this, the problem will be represented in terms of the BM network. The BM occupies a state with probability inversely proportional to the exponential of its energy as indicated by Eq. (1), i.e., lowest energy states are visited most frequently. So, we need to define the energy associated with a cut such that it is inversely proportional to the cost. The problem can be formally stated as follows.[18]

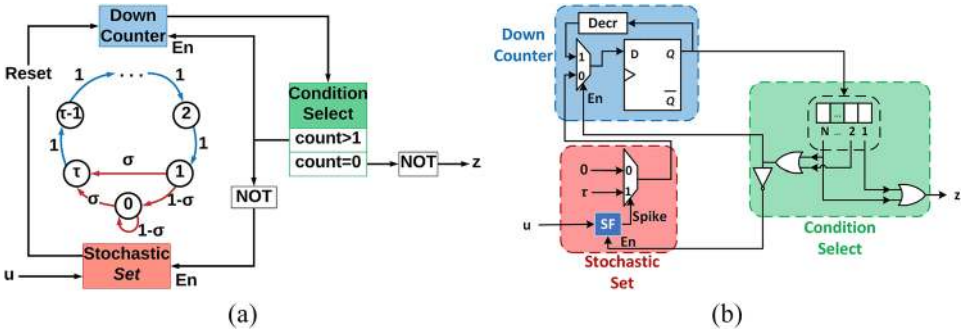

FIG. 2. (a) Block diagram of the Markov chain mathematical model for a neuron is divided into 3 subparts: stochastic *set* (red) models stochastic spiking; down counter (blue) models the refractory period; condition select (green) decides the output of the neuron depending on the current state of neuron. (b) The circuit level implementation consists of the stochastic function (SF) block in addition to the deterministic logic.

Given a graph G = (V, E) with weighted edge set E and vertex set V, find a partition of vertices into disjoint sets S and S' so that the cost function $f(\vec{x})$ defined in Eq. (4) is maximized [Figs. 1(a) and 1(b)],

$$f(\vec{x}) = \sum_{i=1}^{N}\sum_{j=i+1}^{N} w_{ij} * ((1-x_i) * x_j + (1-x_j) * x_i), \tag{4}$$

where

$$x_i = \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{if } i \in S' \end{cases},$$

w_{ij} = weight of the edge between node i and j,

$w_{ii} = 0$.

The above cost function is rearranged in (5) to highlight its similarity to energy of a Boltzmann network shown in (2),

$$f(\vec{x}) = \sum_{i=1}^{N}\Big(\sum_{j=1}^{N} w_{ij}\Big) * x_i + \sum_{i,j}(-2w_{ij}) * x_i * x_j. \tag{5}$$

Therefore, if biases and weights of a Boltzmann network are defined as (6) and (7), respectively, then the cost of the Max-Cut can be mapped to the energy of the BM,

$$bias(i) = \sum_{j=1}^{N} w_{ij}, \tag{6}$$

$$weight(i,j) = -2 * w_{ij}. \tag{7}$$

Once the Max-Cut problem is mapped to the network representing the BM, the BM will settle into the solution where x_i will take values of 0 or 1, indicating whether they are in S or S' [Fig. 1(c)]. The most visited state will be the solution presented by the network.

III. HARDWARE IMPLEMENTATION OF BOLTZMANN MACHINE

The hardware SNN (Stochastic Neural Network) based BM consists of the synapses and neurons. Analog hardware synapses in crossbars [Fig. 1(c)] have been extensively investigated.[20] Hence, we focus on the neuron in this paper. The Markov chain neuron model is shown in Fig. 2(a), and the corresponding hardware implementation is shown in Fig. 2(b). The deterministic transitions in the Markov chain model are replicated using a down counter block. While the down counter is enabled, the state of the neuron (represented by "count") is obtained from the decrementer. Here, the count decreases by 1 after each clock cycle. This deterministic countdown of τ steps acts as the refractory period since the neuron output remains high and no new spike can occur during that time. The condition select block puts various checks on the current state and decides two things depending on the outcome of these checks—(i) output of the neuron and (ii) enable signals of other two blocks, i.e., down counter and stochastic *set* block. The output of the neuron is 0 when count is 0, while it is 1 for all other counts. The down counter is enabled when count is greater than 1, i.e., during the refractory period. Otherwise, the stochastic *set* block is enabled. Here, input u for a neuron controls stochastic switching in the stochastic function *SF* sub-block to produce a sigmoid switching probability.

Two possibilities have been considered for the stochastic function (*SF*) sub-block—(i) pure CMOS and (ii) hybrid (RRAM integrated with CMOS) designs. First, we use a pure CMOS design based Linear Feedback Shift Register (LFSR) to generate pseudorandom numbers on clock—which consists of large digital blocks. Second, we use the hybrid design, where the stochastic switching in the compact RRAM is utilized by a CMOS design. Hence, we will study stochastic RRAM switching experimentally in Sec. III A.

A. PCMO RRAM stochasticity

1. PCMO RRAM device experimental setup

The $Pr_xCa_{1-x}MnO_3$ (x = 0.7) based RRAM devices were fabricated on a 4″ Si substrate. The bottom electrode of Ti (50 nm)/Pt (25 nm) is deposited by sputtering on thermally grown SiO_2 (30 nm). Furthermore, PCMO (65 nm) is deposited by the RF sputtering process at room temperature. Then, the PCMO is crystallized by annealing the sample in N_2 ambient at 650 °C by rapid thermal annealing. After that, the devices were obtained by defining via holes of 1 μm in SiO_2 by electron beam lithography (EBL). Finally, tungsten (W) contact pads (25 μm × 25 μm) are created by sputtering and lift-off of tungsten.[55] The device schematic [Fig. 3(a)] shows the PCMO sandwiched between W and Pt. For the characterization, the voltage is applied between the W and Pt. The DCIV is taken using the Agilent B1500 semiconductor parameter analyzer's Source Measure Unit (SMU) and transient IV by the Waveform Generator/Fast Measure Unit (WGFMU). All the measurements are carried out at room temperature.

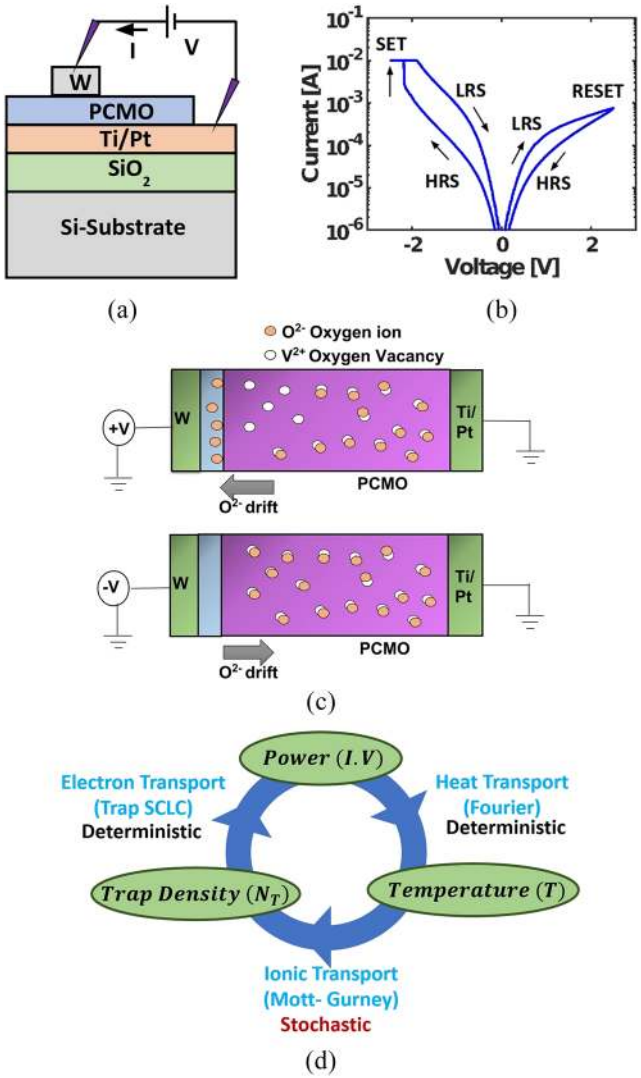

FIG. 3. (a) PCMO thin-film between two metal electrodes (W and Pt) forms the RRAM device. (b) DC IV shows *set* and *reset* transitions. (c)The pictorial representation of *set* and *reset* operation in the PCMO RRAM device in the presence of applied bias shows the movement of oxygen ions to and from the W contact to modulate trap density (N_T), thus producing resistance change. (d) Positive feedback between current (I), temperature (T), and ionic motion [i.e., trap density (N_T)] leads to current shoot-up in the *set* current. The feedback loop shows the corresponding equations. The transport of a few ions given by the Mott-Gurney equation is the origin stochasticity. Such a motion of a few ions locally changes local trap density to initiate positive feedback which produces stochastic *set*.

2. Physics of stochasticity

The typical DC IV characteristics of PCMO RRAM are shown in Fig. 3(b). At low bias, the device does not change its resistance state. The trap density (N_T) dependent space charge limited current (trap-SCLC) flows through the device[56] where $I_{trapSCLC} \propto 1/N_T$. On the application of positive polarity voltage exceeding a voltage threshold, the device switches from a low resistance state (LRS) to a high resistance state (HRS). This is the *reset* operation. Similarly, on the negative polarity exceeding a threshold voltage, the device switches from HRS to LRS. This is the *set* operation. This *set* operation and *reset* operation in the device are attributed to the movement of oxygen ions to and from the PCMO thin film toward the tungsten electrode [Fig. 3(c)], which modulates the resistance through trap density change. The process is described briefly below.

In the *reset* operation, the oxygen ions (O^{2-}) move from bulk toward the tungsten (W) electrode at positive polarity. This O^{2-} egress from PCMO creates oxygen vacancies in the PCMO (i.e., increases the N_T). The movement of ions can be given by the Mott-Gurney equation [Eq. (8)],

$$V_{drift} = afe^{-\frac{E_m}{kT}}e^{-\frac{E}{E_0}} => N_T, \tag{8}$$

where a is the hopping distance, f is the escape frequency, E_m is the activation barrier, E is the electric field, and $E_0 = kT/qa$ is the characteristic electric field.

The increase in N_T in the device increases the resistance and hence leads to reduction in the current which is consistent with the trap-SCLC behavior [Eq. (9)]. The SCLC current depends upon N_T, T, and voltage (V) [i.e., $I_{SCLC}(N_T, T, V)$],

$$I_{trapSCLC} = \frac{I_{trapfree}}{\frac{N_T}{N_V}\exp\left(\frac{E_T - E_V}{k_B T}\right)} \propto \frac{1}{N_T}, \tag{9}$$

$$I_{trapfree} = \frac{9}{8}\mu\varepsilon\left(\frac{V^2}{L^3}\right), \tag{10}$$

where N_V is the effective density of states of the valence band, E_T is the trap energy level, E_V is the valence band energy level, and k_B is the Boltzmann constant. As the voltage is increased further, N_T keeps increasing leading the device into higher resistance states.[57,58]

During *set*, as the negative bias is at W, the oxygen ions (O^{2-}) move away from the electrode and into the PCMO bulk. These ions annihilate the oxygen vacancies in the device, leading to reduction in N_T and hence decrease in resistance. As the resistance is decreased, more current flows through the device. The increase in current leads to Joule heating in the device to further enhance the ionic motion [Eq. (11)]. The device temperature (T) is a function of current and voltage [i.e., $T_{device}(V, I)$],

$$-k\frac{d^2T}{dx^2} + C_V\frac{dT}{dt} = \frac{I.V}{volume}, \tag{11}$$

where k is the thermal conductivity of PCMO, C_V is the specific heat capacity, and volume = area × thickness.

The ionic motion reduces trap density which further increases current. Thus, a positive feedback is developed between current, temperature, and ions, which leads to current shoot-up until a compliance is reached.[57] The *set* dynamics flowchart [Fig. 3(d)] shows the positive feedback loop between heat transport (I → T), ionic transport (T → N_T), and the electron transport (N_T → I). Here, both the heat and electron transport [Eqs. (11) and (9), respectively] are deterministic processes, whereas the ionic transport [Eq. (8)] is stochastic in nature, as indicated in Fig. 3(d). The stochasticity in the ionic transport comes from the hopping probability associated with the oxygen ions. The transport of a few ions modifies the potential profile locally for current transport and modulates the DC current and related heating. In the *set* process, the transport of a few ions

may initiate positive feedback of current and heating locally. This local hot spot may spread to the entire PCMO layer—producing a stochastic *set* process. This leads to the stochastic nature of current switching when observed in the transient measurements [Fig. 3(b)]. The probability of switching is voltage-dependent, i.e., it is zero at low bias and increases and saturates to 1 at high bias.

B. Implementation of stochastic function (*SF*) block

The stochastic function (*SF*) block in the block diagram [Fig. 2(b)] is the most challenging element of the neuron. We compare the three different implementations shown in Fig. 4.

The first two designs are a Digital Precision-controlled Sigmoid (DPS) implementation, as shown in Fig. 4, to enable high bit-precision based replication of sigmoid using a Lookup Table (LUT). Both designs require an input preprocessing stage, where the analog input signal from the crossbar array of weights (u) is sampled by ADC. For a Pure CMOS based implementation [Fig. 4(a)], the digital signal is then processed through the Lookup Table (LUT), which outputs a threshold value. An LFSR generates a pseudorandom number. In the readout stage, the comparison of the LFSR output with the LUT output determines whether the neuron has spiked or not, i.e., if LFSR output exceeds the LUT output, then the neuron has spiked, else not. For a DPS hybrid scheme [Fig. 4(b)], the LUT translates the input of the neuron to a digital voltage value to be applied to the RRAM to enable stochastic switching corresponding to the probability $\sigma(u)$. This digital voltage value is converted by the DAC to a voltage to be applied to the RRAM. This produces a high/low output in readout stage depending on the state of RRAM (low or high resistance state). If the RRAM has switched to low resistance, a *reset* bias is applied to *reset* it during the countdown and get it prepared for the next switching.

The implementations described above fail to utilize an important property of PCMO RRAM, i.e., the approximately sigmoidal switching probability. Alternatively, we implement the Approximate Analog Sigmoid (AAS) schemes that the naturally approximate sigmoidal switching of RRAM is utilized directly [Fig. 4(c)]. Here, we directly give the input voltage (u) from the crossbar to the RRAM such that $\sigma(u)$ based probabilistic switching is obtained. It requires a scale-and-shift operation, which is managed by the operational amplifier that is present in all 3 designs. Thus, the ADC and DAC operations are avoided. The stochastic switching and state readout stages are identical to the second (i.e., Hybrid DPS) design.

IV. EXPERIMENTAL RESULTS

A. Nanoscale stochastic switching element: RRAM data measurement

As discussed in Fig. 3, the PCMO RRAM enables a voltage dependent probabilistic current switching during set. To experimentally study this stochasticity, the transient current is measured for a given set pulse amplitude to observe the switching time, i.e., the time at which current shoots up. The transient current is

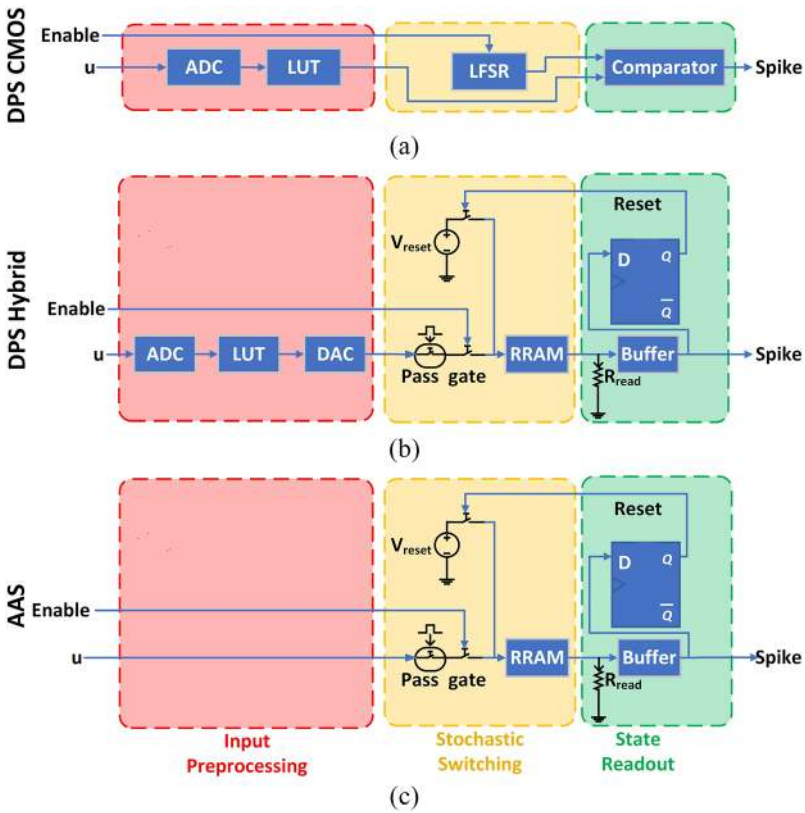

FIG. 4. Three implementations of the SF block, which is divided in 3 processing stages: input preprocessing, stochastic switching, and state readout. (a) Pure CMOS based DPS scheme uses LFSR as the random number generator. (b) Hybrid scheme uses RRAM as the stochastic element and Lookup Table (LUT) to generate the required *set* voltage. (c) Proposed AAS scheme connects input directly to RRAM taking advantage of the similarity in shape of switching probability to sigmoid activation. Thus, the ADC and DAC circuits in the input preprocessing stage are avoided.

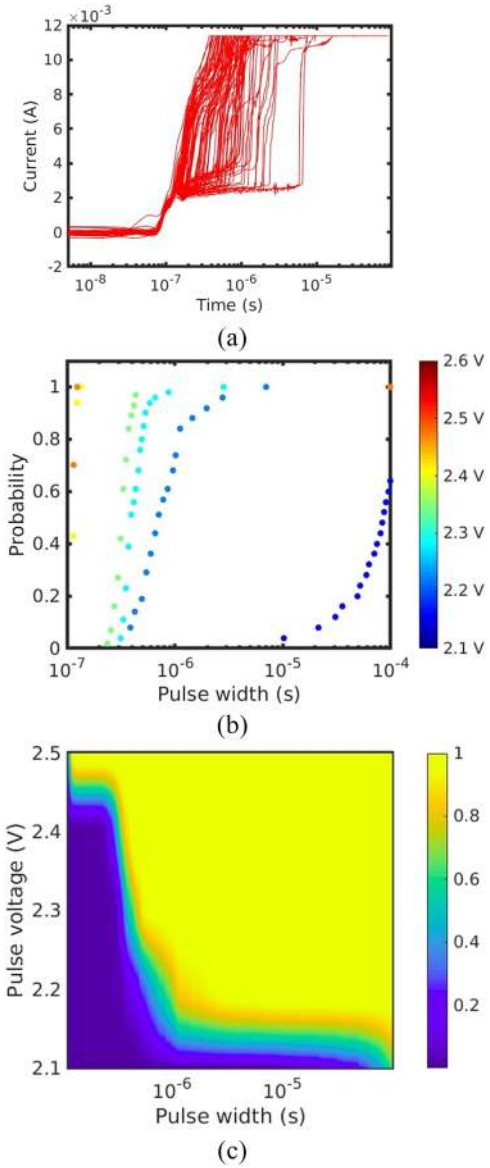

FIG. 5. (a) RRAM current transient at 2.2 V pulse for 1 ms shows the stochastic *set*. (b) The probability of RRAM switching at various voltages shows stochastic switching control where the colorbar shows applied pulse voltage. (c) Probability of RRAM switching as a function of pulse width and voltage where the colorbar shows the probability of switching.

repeatedly measured during consecutive set measurements by alternatively applying a reset and a set pulse of duration 1 ms. One such example is shown in Fig. 5(a). The stochastic switching in time can be observed for a fixed set voltage pulse (−2.2 V/1 ms) for 100 runs. The current shoots up at different time instants for different set runs giving a probability distribution in the switching time. This stochasticity is further modulated by the applied voltage pulse amplitude and duration. We plot the cumulative probability distribution (CDF) of switching time in Fig. 5(b) for different applied voltages. When the voltage pulse amplitude is very high [>2.4 V, orange and red curves in Fig. 5(b)], the switching is deterministic, i.e., the time to set is a very narrow distribution in time. As the voltage pulse amplitude is decreased [<2.4 V, green and blue curves in Fig. 5(b)], the device switching becomes more stochastic, i.e., the time to set is a broad distribution in time. The probability of switching of RRAM by a pulse of a specific amplitude and pulse width is extracted out of the experimental CDF data by interpolation to generate the contour plot shown in Fig. 5(c).

We plot the switching probability as a function of pulse amplitude for three different pulse widths in Fig. 6(a) using the contour plot. Figure 6(b) shows that the RRAM switching probability function at a fixed pulse width closely resembles a sigmoid function after a linear transformation to the voltage axis. In Fig. 7(a), we have shown how analog u is converted to stochastic digital spikes through an RRAM. The DPS implementation needs to convert analog u to digital form through an ADC. The error reduces as bit precision

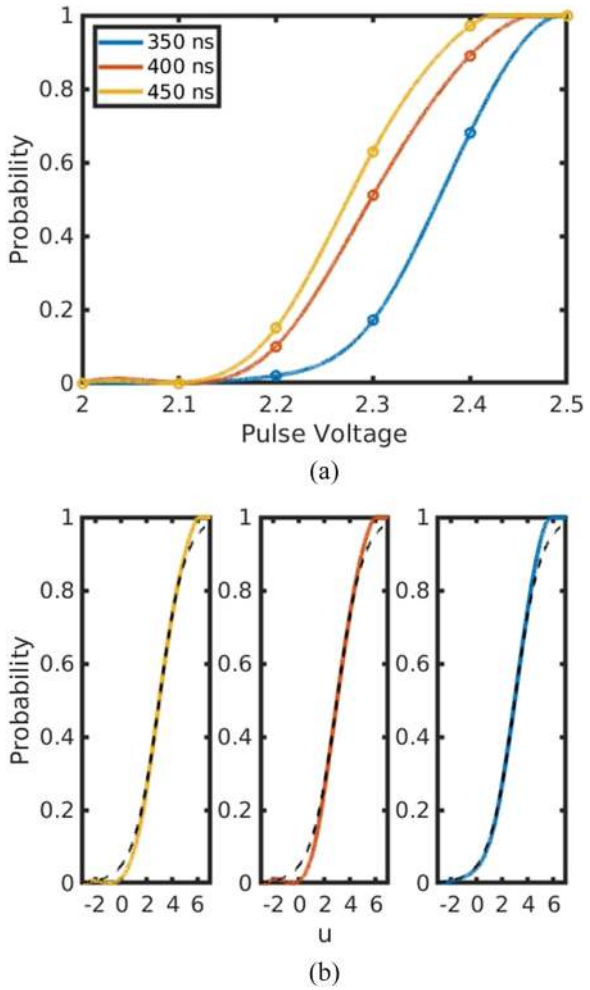

FIG. 6. (a) Probability of RRAM switching at three different pulse widths. (b) The RRAM switching probability at these pulse widths is compared to a sigmoid function (black) after linear transformation (shift and scale operation).

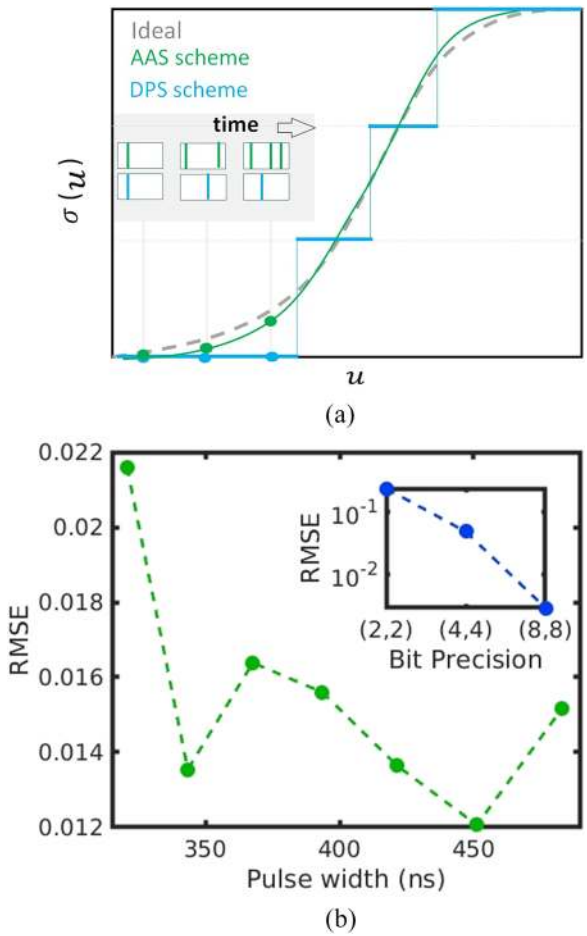

FIG. 7. (a) The ideal sigmoid $\sigma(u)$ may be digitized in the Digital Precision Scheme (DPS) or approximated to analog in the Approximate Analog Scheme (AAS) where u is analog to preserve the advantages of analog matrix multiplication.[17] The inset shows that AAS (green) is able to approximate the ideal $\sigma(u)$, providing digital spikes changing with u over the entire range of analog u. In comparison, the output digital spikes are unchanged for large ranges of u depending upon the bit-precision. (b) Error dependence on pulse-width in the AAS scheme is rather constant, while for the DPS scheme, error reduces with bit precision increase (inset). The root mean square error values at the 3 pulse widths are 1.35%, 1.56%, and 1.21%, respectively. A maximum error of 3.5% was observed over the complete range of pulse widths in the AAS scheme. The inset shows that the DPS error decreases with an increase in bit-precision.

increases for DPS in the inset of Fig. 7(b). In comparison, the AAS implementation needs no such conversion. It applies the analog u to the RRAM to obtain digital stochastic switching where the probability of switching is modulated in an analog fashion. Although the probability of switching is analog, the RRAM switching is digital because the switching from high to low resistance is abrupt with a large (10x) decrease in resistance—akin to digital low ("0") to high ("1") state. A voltage divider is designed with R_{read} in series to ensure that the voltage change is compatible with the buffer in Fig. 4(b). The neuron is also stochastic as the RRAM has probabilistic switching. Thus, the RRAM input is an analog voltage, the switching is digital, but the dependence of the probability of switching on input voltage is analog—which mimics a sigmoid. That is why this implementation is termed approximate analog sigmoid.

The pulse width gives us control over reducing the error between the functional form of the probability vs voltage amplitude curve and an exact sigmoid function. Given various pulse-widths, we observe a minimum rms error of 1.21% compared to an ideal sigmoid for a pulse width of 450 ns [Fig. 7(b)]. Thus, we can implement a near-sigmoid probability function using an RRAM by linearly transforming the input membrane potential, u. This transformation can be accommodated as a part of the weight scaling in the crossbar array. The resultant voltage can then be applied as a pulse to the RRAM to implement approximate sigmoid probability activation, thus eliminating the need for lookup tables and preprocessing blocks. The combined implementation of sigmoidal stochasticity vs applied pulse amplitude to PCMO RRAM for a fixed pulse width is the key enabler for AAS architecture.

As discussed earlier in Sec. III A 2 titled "Physics of Stochasticity," the origin of stochasticity is based on the motion of a few ions to kick-start the positive feedback process that produces the HRS to LRS transition. Hence, device area scaling will reduce the number of ions required to be transported and enhance number fluctuation. This requires further investigation.

B. Performance, Area, and Energy Consumption

As mentioned earlier, the network of neurons should sample from Boltzmann distribution. Figure 8(a) shows a small sample of joint distribution of 4 neurons (i.e., 2^4 cases) of a 10 neuron system with 2^{10} states. For comparison, we consider the result obtained from Gibbs sampling as the baseline to show that the network is closely sampling from the Boltzmann distribution. However, the ultimate demonstration is the solution of an NP hard problem, which is presented below.

The solutions of the Max-Cut problem are compared using the relative cost metric. The relative cost is defined as the ratio of the solution given by the network in a run to the optimal solution. Thus, a relative cost closer to 1 would mean a better solution. The performance on the 125 node Max-Cut solution is evaluated by simulations for the three schemes: AAS and DPS schemes (hybrid and pure CMOS). The DPS scheme performance was presented for different bit-precision cases defined as (u-bit, o-bit). Here, u-bit is the resolution of the input (u) and output bit (o-bits) defines the probability resolution for the pure CMOS scheme or voltage-resolution for the hybrid scheme. Thus, 5 cases were simulated: (4, 4), (5, 6), (6, 6), (6, 8), (8, 8).

To evaluate the circuit density, standard designs of digital components[59] are used for estimation at 65 nm technology, while representative mixed signal component performances for ADC and DAC (in the same technology) are directly taken from the literature[60,61] for a 1 MHz circuit operation for all three cases. A circuit area is estimated for the circuit design.

For digital components, the switching power is computed using the expression

$$P = 0.5\alpha f C V^2, \tag{12}$$

where α is the switching probability of a transistor, f is the operating frequency, C is the output load capacitance, and V is the supply

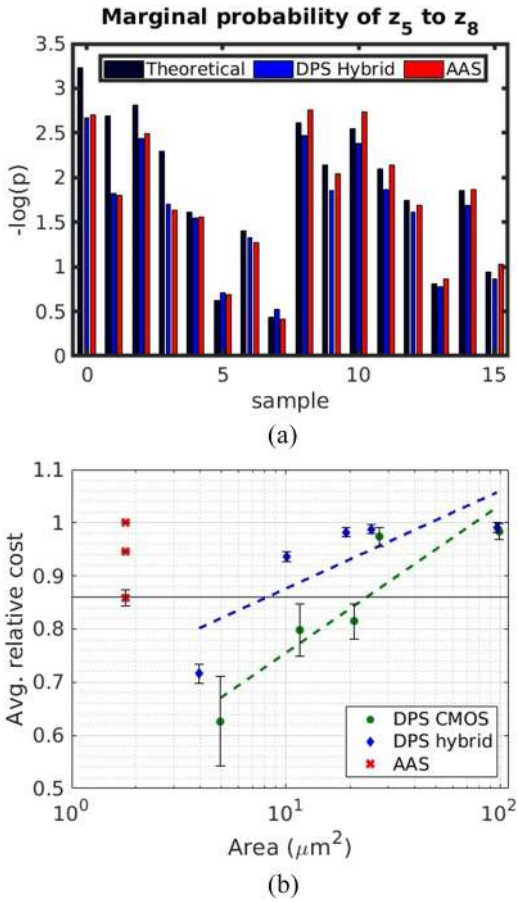

FIG. 8. (a) The probability of occurrence of the 16 possible states (states of neurons z_5 to z_8 in a 10 node BM) is compared for 3 different modes: (i) theoretical probability calculated from the Boltzmann formula, (ii) DPS hybrid scheme at (8, 12) precision, and (iii) AAS scheme at a fixed pulse width of 368 ns. (b) The performance (relative cost) vs circuit area is compared with the lower limit for the Goemans-Williamson algorithm of the Max-Cut problem. The AAS scheme shows performance reduction with sigmoid error-enhancement in the range of 1.5%–3%. The DPS scheme reduces in performance and circuit area with the reducing precision in 5 cases: (8, 8), (6, 8), (6, 6), (5, 6), (4, 4). Overall, the AAS scheme has 10× lower circuit area at equivalent performance compared to the DPS scheme.

voltage. $0.5CV^2$ is the energy dissipation in a single high-to-low or low-to-high transition. For the ADC and DAC, the powers and conversion times have been taken from Refs. 60 and 61, which are state of the art in the literature with respect to the energy-per-bit and the typical resolution.

Figure 8(b) shows the performance of network vs area of circuit comparison. The solid line indicates the performance of the Goemans-Williamson algorithm for Max-Cut. The performance reduces with bit precision reduction for DPS designs as the neuronal area reduces—indicating a trade-off. The DPS neuron's area is dominated by the mixed signal components (ADC and DAC)—whose sizes are related to the precision. Thus, the reduction in size comes at the cost of reduced performance. The hybrid scheme appeared more resilient than the pure CMOS scheme. For the AAS scheme, the three different pulse-widths (in the 300–450 ns range) producing different errors (in the range of 1.5%–3%) compared to ideal sigmoid were simulated. The accuracy of the sigmoidal approximation of stochasticity depends upon the choice of pulse time. Figure 7(b) shows minima in accuracy at 450 ns which is 1.83× smaller than 350 ns. For the AAS scheme, keeping the pulse width 350 ns instead of 450 ns gives 1.32× energy reduction. This reduction occurs due to a decrease in energy dissipation through RRAM. However, it is insignificant for hybrid DPS where preprocessing is dominant. The BM performance increased with sigmoidal error reduction. The AAS design has the same area for 3 different pulse-widths, while the related error determines performance. Furthermore, we compare the performance with the Goemans-Williamson algorithm[62] that guarantees relative cost better than 0.87× of the optimal. The AAS and high-precision DPS results are significantly better than this lower bound. Overall, the AAS design occupies 1/10th the area compared to both the DPS designs at greater than equivalent performance of 98%–100%. In fact, an AAS circuit at 450 ns pulse width performs even better than both the (8,8) bit precision DPS circuits which occupy close to 50× more area.

In terms of energy per spike for a neuron, the energy dissipated by various components and the total energy dissipated are shown in Figs. 9(a) and 9(b). The direct RRAM scheme dissipates 1/4th the energy of DPS-pure CMOS scheme and 1/9th the energy of DPS-hybrid scheme. This energy saving is attributed to the elimination of the preprocessing stage which includes the ADC and DAC

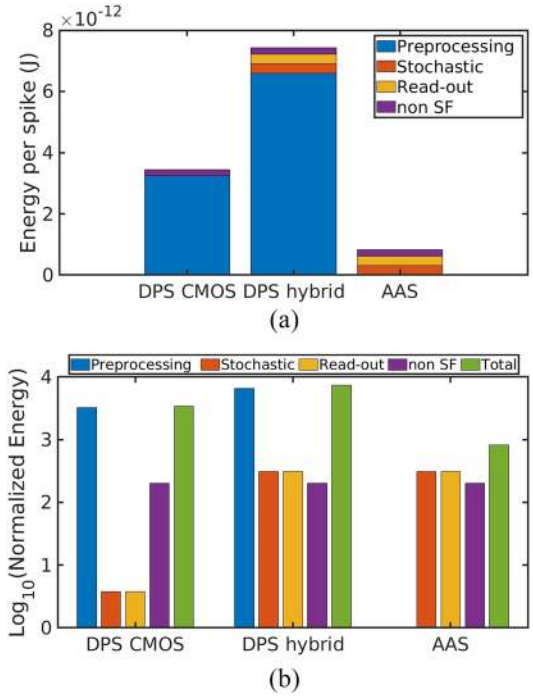

FIG. 9. The DPS schemes have bit-precision of (8, 8) for CMOS and hybrid schemes. (a) Estimated total energy dissipation per spike for each design is distributed into various blocks. (b) Estimated energy dissipation per spike by each part of circuit normalized with respect to 1 fJ.

components. While present results are promising, the effect on the nature of stochasticity due to scaling PCMO RRAM needs to be evaluated to estimate the effect on system performance. Furthermore, device-to-device variability in the stochasticity of the neuron may have a significant impact on performance to require device-system co-design.

V. CONCLUSION

In this paper, we study stochastic neuron design for the BM to solve classic NP hard problems, which are of great theoretical interest and with a wide range of practical applications. We show that the PCMO RRAM has an approximately analog sigmoidal (AAS) stochastic switching experimentally. We utilize this property to design stochastic neurons to solve a Max-Cut problem—a typical NP hard problem. A comparison to the digital precision-controlled scheme (DPS) by pure and hybrid CMOS is performed. All schemes perform better than heuristic Goemans-Williamson algorithm limits. We show that the AAS scheme has a 4× power and 10× area reduction over DPS schemes as well as the origins of the improvement. Thus, PCMO RRAM based stochastic neurons are highly promising for hardware BMs—an example of stochastic neuromorphic computing, to solve NP hard problems, which are extremely challenging for von Neumann computing.

ACKNOWLEDGMENTS

The work was partially funded by DST Nano Mission and Ministry of Electronics and IT (MeitY). It was performed at the IIT Bombay Nanofab Facility. S.L. was funded by Intel Ph.D. Fellowship and Visveswaraya Fellowship. V.S. was funded through the Prime Minister's Research Fellowship (PMRF).

REFERENCES

[1] G. Indiveri and S. C. Liu, "Memory and information processing in neuromorphic systems," Proc. IEEE **103**(8), 1379–1397 (2015).
[2] N. Kasabov, N. Sengupta, and N. Scott, "From von Neumann, John Atanasoff and ABC to neuromorphic computation and the NeuCube spatio-temporal data machine," in *2016 IEEE Proceedings of the 8th International Conference on Intelligent Systems, IS 2016* (IEEE, 2016), pp. 15–21.
[3] J. Ramanujam and P. Sadayappan, "Optimization by neural networks," in *IEEE 1988 International Conference on Neural Networks, July 1988* (IEEE, 1988), Vol. 2, pp. 325–332.
[4] D. G. Bounds, "New optimization methods from physics and biology," Nature **329**(6136), 215–219 (1987).
[5] S. M. Graham, A. Joshi, and Z. Pizlo, "The traveling salesman problem: A hierarchical model," Mem. Cognit. **28**(7), 1191–1204 (2000).
[6] S. Habenschuss, Z. Jonke, and W. Maass, "Stochastic computations in cortical microcircuit models," PLoS Comput. Biol. **9**(11), e1003311 (2013).
[7] K. A. Smith, "Neural networks for combinatorial optimization: A review of more than a decade of research," INFORMS J. Comput. **11**(1), 15–34 (1999).
[8] W. Maass, "Networks of spiking neurons: The third generation of neural network models," Neural Networks **10**(9), 1659–1671 (1997).
[9] H. S. Seung, "Learning in spiking neural networks by reinforcement of stochastic synaptic transmission," Neuron **40**(6), 1063–1073 (2003).
[10] M. D. McDonnell and D. Abbott, "What is stochastic resonance? Definitions, misconceptions, debates, and its relevance to biology," PLoS Comput. Biol. **5**(5), e1000348 (2009).
[11] Z. Jonke, S. Habenschuss, and W. Maass, "Solving constraint satisfaction problems with networks of spiking neurons," Front. Neurosci. **10**(118), 3 (2016).
[12] M. J. Berry and M. Meister, "Refractoriness and neural precision," J. Neurosci. **18**(6), 2200–2211 (1998).
[13] G. A. Fonseca Guerra and S. B. Furber, "Using stochastic spiking neural networks on spinnaker to solve constraint satisfaction problems," Front. Neurosci. **11**, 714 (2017).
[14] S. Kumar, J. P. Strachan, and R. S. Williams, "Chaotic dynamics in nanoscale NbO_2 Mott memristors for analogue computing," Nature **548**, 318 (2017).
[15] A. Parihar, N. Shukla, M. Jerry, S. Datta, and A. Raychowdhury, "Vertex coloring of graphs via phase dynamics of coupled oscillatory networks," Sci. Rep. **7**(1), 911 (2017).
[16] S. Lashkare, P. Kumbhare, V. Saraswat, and U. Ganguly, "Transient joule heating-based oscillator neuron for neuromorphic computing," IEEE Electron Device Lett. **39**(9), 1437–1440 (2018).
[17] M. N. Bojnordi and E. Ipek, "Memristive Boltzmann machine: A hardware accelerator for combinatorial optimization and deep learning," in *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)* (IEEE, 2016), pp. 1–13.
[18] J. H. Korst and E. H. Aarts, "Combinatorial optimization on a Boltzmann machine," J. Parallel Distrib. Comput. **6**(2), 331–357 (1989).
[19] L. Buesing, J. Bill, B. Nessler, and W. Maass, "Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons," PLoS Comput. Biol. **7**(11), e1002211 (2011).
[20] D. Kuzum, S. Yu, and H. S. Philip Wong, "Synaptic electronics: Materials, devices and applications," Nanotechnology **24**(38), 382001 (2013).
[21] R. A. Nawrocki, R. M. Voyles, and S. E. Shaheen, "A mini review of neuromorphic architectures and implementations," IEEE Trans. Electron Devices **63**(10), 3819–3829 (2016).
[22] G. Indiveri, B. Linares-Barranco, T. J. Hamilton, A. van Schaik, R. Etienne-Cummings, T. Delbruck, S. C. Liu, P. Dudek, P. Häfliger, S. Renaud, J. Schemmel, G. Cauwenberghs, J. Arthur, K. Hynna, F. Folowosele, S. Saighi, T. Serrano-Gotarredona, J. Wijekoon, Y. Wang, and K. Boahen, "Neuromorphic silicon neuron circuits," Front. Neurosci. **5**, 73 (2011).
[23] R. Sarpeshkar, "Analog versus digital: Extrapolating from electronics to neurobiology," Neural Comput. **10**(7), 1601–1638 (1998).
[24] S. Dutta, V. Kumar, A. Shukla, N. R. Mohapatra, and U. Ganguly, "Leaky integrate and fire neuron by charge-discharge dynamics in floating-body MOSFET," Sci. Rep. **7**(1), 8257 (2017).
[25] S. Dutta, T. Bhattacharya, N. R. Mohapatra, M. Suri, and U. Ganguly, "Transient variability in SOI-based LIF neuron and impact on unsupervised learning," IEEE Trans. Electron Devices **65**(11), 5137–5144 (2018).
[26] S. Gaba P. Sheridan, J. Zhou, S. Choi, and W. Lu, "Stochastic memristive devices for computing and neuromorphic applications," Nanoscale **5**(13), 5872–5878 (2013).
[27] S. H. Jo, K. H. Kim, and W. Lu, "Programmable resistance switching in nanoscale two-terminal devices," Nano Lett. **9**(1), 496–500 (2009).
[28] Q. Li, A. Khiat, I. Salaoru, H. Xu, and T. Prodromakis, "Stochastic switching of TiO_2-based memristive devices with identical initial memory states," Nanoscale Res. Lett. **9**(1), 293 (2014).
[29] M. Hu, Y. Wang, W. Wen, Y. Wang, and H. Li, "Leveraging stochastic memristor devices in neuromorphic hardware systems," IEEE J. Emerging Sel. Top. Circuits Syst. **6**(2), 235–246 (2016).
[30] P. Krzysteczko, J. Münchenberger, M. Schäfers, G. Reiss, and A. Thomas, "The memristive magnetic tunnel junction as a nanoscopic synapse-neuron system," Adv. Mater. **24**(6), 762–766 (2012).
[31] M. Hu, Y. Wang, Q. Qiu, Y. Chen, and H. Li, "The stochastic modeling of TiO_2 memristor and its usage in neuromorphic system design," in *Proceedings of the Asia and South Pacific Design Automation Conference ASP-DAC* (IEEE, 2014), pp. 831–836.
[32] T. Tuma, A. Pantazi, M. Le Gallo, A. Sebastian, and E. Eleftheriou, "Stochastic phase-change neurons," Nat. Nanotechnol. **11**(8), 693–699 (2016).
[33] P. Debashis, R. Faria, K. Y. Camsari, and Z. Chen, "Design of stochastic nanomagnets for probabilistic spin logic," IEEE Magn. Lett. **9**, 1–5 (2018).
[34] R. Zand, K. Y. Camsari, S. Datta, and R. F. DeMara, "Composable probabilistic inference networks using MRAM-based stochastic neurons," ACM J. Emerg. Tech. Com. **15**(2), 17 (2019).

[35]O. Hassan, R. Faria, K. Y. Camsari, J. Z. Sun, and S. Datta, "Low-barrier magnet design for efficient hardware binary stochastic neurons," IEEE Magn. Lett. **10**, 1–5 (2019).
[36]W. Yi, K. K. Tsang, S. K. Lam, X. Bai, J. A. Crowell, and E. A. Flores, "Biological plausibility and stochasticity in scalable VO_2 active memristor neurons," Nat. Commun. **9**(1), 4661 (2018).
[37]M. S. Feali and A. Ahmadi, "Realistic Hodgkin–Huxley axons using stochastic behavior of memristors," Neural Process. Lett. **45**(1), 1–14 (2017).
[38]M. Al-Shedivat, R. Naous, E. Neftci, G. Cauwenberghs, and K. N. Salama, "Inherently stochastic spiking neurons for probabilistic neural computation," in *International IEEE/EMBS Conference on Neural Engineering, NER* (IEEE, 2015), Vol. 2015-July, pp. 356–359.
[39]C. Merkel and D. Kudithipudi, "A current-mode CMOS/memristor hybrid implementation of an extreme learning machine," in *Proceedings of the 24th Edition of the Great Lakes Symposium on VLSI, ser. GLSVLSI '14* (ACM, 2014), pp. 241–242.
[40]R. Naous, M. Al-Shedivat, and K. N. Salama, "Stochasticity modeling in memristors," IEEE Trans. Nanotechnol. **15**(1), 15–28 (2016).
[41]M. Al-Shedivat, R. Naous, G. Cauwenberghs, and K. N. Salama, "Memristors empower spiking neurons with stochasticity," IEEE J. Emerging Sel. Top. Circuits Syst. **5**(2), 242–253 (2015).
[42]M. Prezioso, I. Kataeva, F. Merrikh-Bayat, B. Hoskins, G. Adam, T. Sota, K. Likharev, and D. Strukov, "Modeling and implementation of firing-rate neuromorphic-network classifiers with bilayer $Pt/Al_2O_3/TiO_{2-x}/Pt$ memristors," in *Proceedings Technical Digest—International Electron Devices Meeting, IEDM* (IEEE, 2015), pp. 1–17.
[43]P. Wijesinghe, Y. Kim, A. Sengupta, P. Panda, and K. Roy, "Magnetic tunnel junction mimics stochastic cortical spiking neurons," Sci. Rep. **6**(1), 30039 (2016).
[44]J. Cai, B. Fang, L. Zhang, W. Lv, B. Zhang, T. Zhou, G. Finocchio, and Z. Zeng, "Voltage-controlled spintronic stochastic neuron based on a magnetic tunnel junction," Phys. Rev. Appl. **11**(3), 034015 (2019).
[45]P. Wijesinghe, A. Ankit, A. Sengupta, and K. Roy, "An all-memristor deep spiking neural computing system: A step towards realizing the low power, stochastic brain," e-print arXiv:1712.01472 (2017).
[46]D. Garbin, E. Vianello, O. Bichler, Q. Rafhay, C. Gamrat, G. Ghibaudo, B. DeSalvo, and L. Perniola, "HfO_2-based OxRAM devices as synapses for convolutional neural networks," IEEE Trans. Electron Devices **62**(8), 2494–2501 (2015).
[47]X. Liu, K. P. Biju, E. M. Bourim, S. Park, W. Lee, J. Shin, and H. Hwang, "Low programming voltage resistive switching in reactive metal/polycrystalline $Pr_{0.7}Ca_{0.3}MnO_3$ devices," Solid State Commun. **150**(45-46), 2231–2235 (2010).
[48]N. Panwar, P. Kumbhare, A. K. Singh, N. Venkataramani, and U. Ganguly, "Effect of morphological change on unipolar and bipolar switching characteristics in $Pr_{0.7}Ca_{0.3}MnO_3$ based RRAM," Mater. Res. Soc. Symp. Proc. **1729**, 47–52 (2015).
[49]S. Park, M. Chu, J. Kim, J. Noh, M. Jeon, B. Hun Lee, H. Hwang, B. Lee, and B. G. Lee, "Electronic system with memristive synapses for pattern recognition," Sci. Rep. **5**, 10123 (2015).
[50]N. Panwar, B. Rajendran, and U. Ganguly, "Arbitrary spike time dependent plasticity (STDP) in memristor by analog waveform engineering," IEEE Electron Device Lett. **38**(6), 740–743 (2017).
[51]S. Lashkare, N. Panwar, P. Kumbhare, B. Das, and U. Ganguly, "PCMO-based RRAM and NPN bipolar selector as synapse for energy efficient STDP," IEEE Electron Device Lett. **38**(9), 1212–1215 (2017).
[52]A. V. Babu, S. Lashkare, U. Ganguly, and B. Rajendran, "Stochastic learning in deep neural networks based on nanoscale PCMO device characteristics," Neurocomputing **321**, 227–236 (2018).
[53]S. Lashkare, S. Chouhan, T. Chavan, A. Bhat, P. Kumbhare, and U. Ganguly, "PCMO RRAM for integrate-and-fire neuron in spiking neural networks," IEEE Electron Device Lett. **39**(4), 484–487 (2018).
[54]R. S. Wang and L. M. Wang, "Maximum cut in fuzzy nature: Models and algorithms," J. Comput. Appl. Math. **234**(1), 240–252 (2010).
[55]P. Kumbhare and U. Ganguly, "Ionic transport barrier tuning by composition in $Pr_{1-x}Ca_xMnO_3$-based selector-less RRAM and its effect on memory performance," IEEE Trans. Electron Devices **65**(6), 2479–2484 (2018).
[56]L. E. Lager and G. Mur, "Least-squares minimising finite-element formulation for static and stationary electric and magnetic fields," IEEE Trans. Magn. **34**(5), 2419–2424 (1998).
[57]A. Khanna, S. Prasad, N. Panwar, and U. Ganguly, "Reaction-drift model for switching transients in $Pr_{0.7}Ca_{0.3}MnO_3$ based resistive RAM," e-print arXiv:1612.05293 (2016).
[58]N. Panwar and U. Ganguly, "Temperature effects in set/reset voltage–time dilemma in $Pr_{0.7}Ca_{0.3}MnO_3$-based RRAM," IEEE Trans. Electron Devices **66**(1), 829–832 (2019).
[59]R. J. Tocci, *Digital Systems: Principles and Applications*, 7th ed. (Prentice Hall Professional Technical Reference, 1980).
[60]K. Chander and S. Choudhry, "65nm low power digital to analog converter for CUWB," in *Proceedings of the 2nd International Conference on Trends in Electronics and Informatics, ICOEI 2018* (IEEE, 2018), Vol. 7, No. 1, pp. 610–614.
[61]G. Yin, H. G. Wei, U. F. Chio, S. W. Sin, U. Seng-Pan, Z. Wang, and R. P. Martins, "A 0.024 mm^2 4.9 fJ 10-bit 2 MS/s SAR ADC in 65 nm CMOS," in *European Solid-State Circuits Conference* (IEEE, 2012), pp. 377–380.
[62]M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," J. Assoc. Comput. Mach. **42**(6), 1115–1145 (2002).