Implementation of Ternary Weights With Resistive RAM Using a Single Sense Operation per Synapse

Axel Laborieux, Marc Bocquet, Hirtzlin Tifenn, Jacques-Olivier Klein, Etienne Nowak, Elisa Vianello, Jean-Michel Portal, Damien Querlioz

Implementation of Ternary Weights with Resistive RAM Using a Single Sense Operation per Synapse

Axel Laborieux, *Student Member, IEEE,* Marc Bocquet, Tifenn Hirtzlin, *Student Member, IEEE,* Jacques-Olivier Klein, *Member, IEEE,* Etienne Nowak, Elisa Vianello, *Member, IEEE,* Jean-Michel Portal *Member, IEEE,* and Damien Querlioz *Member, IEEE*

***Abstract*—The design of systems implementing low precision neural networks with emerging memories such as resistive random access memory (RRAM) is a significant lead for reducing the energy consumption of artificial intelligence. To achieve maximum energy efficiency in such systems, logic and memory should be integrated as tightly as possible. In this work, we focus on the case of ternary neural networks, where synaptic weights assume ternary values. We propose a two-transistor/two-resistor memory architecture employing a precharge sense amplifier, where the weight value can be extracted in a single sense operation. Based on experimental measurements on a hybrid 130 nm CMOS/RRAM chip featuring this sense amplifier, we show that this technique is particularly appropriate at low supply voltage, and that it is resilient to process, voltage, and temperature variations. We characterize the bit error rate in our scheme. We show based on neural network simulation on the CIFAR-10 image recognition task that the use of ternary neural networks significantly increases neural network performance, with regards to binary ones, which are often preferred for inference hardware. We finally evidence that the neural network is immune to the type of bit errors observed in our scheme, which can therefore be used without error correction.**

***Index Terms*—Neural Networks, Resistive Memory, Quantized Neural Networks, Low Voltage Operation, Sense Amplifier.**

I. Introduction

Artificial Intelligence has made tremendous progress in recent years due to the development of deep neural networks. Its deployment at the edge, however, is currently limited by the high power consumption of the associated algorithms [1]. Low precision neural networks are currently emerging as a solution, as they allow the development of low power consumption hardware specialized in deep learning inference [2]. The most extreme case of low precision neural networks, the Binarized Neural Network (BNN), also called XNOR-NET, is receiving particular attention as it is especially efficient for hardware implementation: both synaptic weights and neuronal activations assume only binary values [3], [4]. Remarkably, this type of neural network can achieve high accuracy on vision tasks [5]. One particularly investigated lead is to fabricate hardware BNNs with emerging memories such as resistive RAM or memristors [6]–[13]. The low memory requirements of BNNs, as well as their reliance on simple arithmetic operations, make them indeed particularly adapted for "in-memory" or "near-memory" computing approaches, which achieve superior energy-efficiency by avoiding the von Neumann bottleneck entirely.

Ternary neural networks [14] (TNN, also called Gated XNOR-NET, or GXNOR-NET [15]), which add the value 0 to synaptic weights and activations, are also considered for hardware implementations [16]–[19]. They are comparatively receiving less attention than binarized neural networks, however. In this work, we highlight that implementing TNNs does not necessarily imply considerable overhead with regards to BNNs. We introduce a two-transistor/two-resistor memory architecture for TNN implementation. The array uses a precharge sense amplifier for reading weights, and the ternary weight value can be extracted in a single sense operation, by exploiting the fact that latency of the sense amplifier depends on the resistive states of the memory devices. This work extends a hardware developed for the energy-efficient implementation of BNNs [6], where the synaptic weights are implemented in a differential fashion. We, therefore, show that it can be extended to TNNs without overhead on the memory array.

The contribution of this work is as follows. After presenting the background of the work (section II):

- We demonstrate experimentally, on a fabricated 130 nm RRAM/CMOS hybrid chip, a strategy for implementing ternary weights using a precharge sense amplifier, which is particularly appropriate when the sense amplifier is operated at low supply voltage (section III).
- We analyze the bit errors of this scheme experimentally and their dependence on the RRAM programming conditions (section V).
- We verify the robustness of the approach to process, voltage, and temperature variations (section IV).
- We carry simulations that show the superiority of TNNs over BNNs on the canonical CIFAR-10 vision task, and evidence the error resilience of hardware TNNs (section VI).

Axel Laborieux, Tifenn Hirtzlin, Jacques-Olivier Klein, and Damien Querlioz are with Université Paris-Saclay, CNRS, Centre de Nanosciences et de Nanotechnologies, 91120 Palaiseau, France. email: damien.querlioz@c2n.upsaclay.fr

Marc Bocquet and Jean-Michel Portal are with Institut Matériaux Microélectronique Nanosciences de Provence, Univ. Aix-Marseille et Toulon, CNRS, France.

Etienne Nowak and Elisa Vianello are with Université Grenoble-Alpes, CEA, LETI, Grenoble, France.

This work was supported by the ERC Grant NANOINFER (715872) and the ANR grant NEURONIC (ANR-18-CE24-0009).

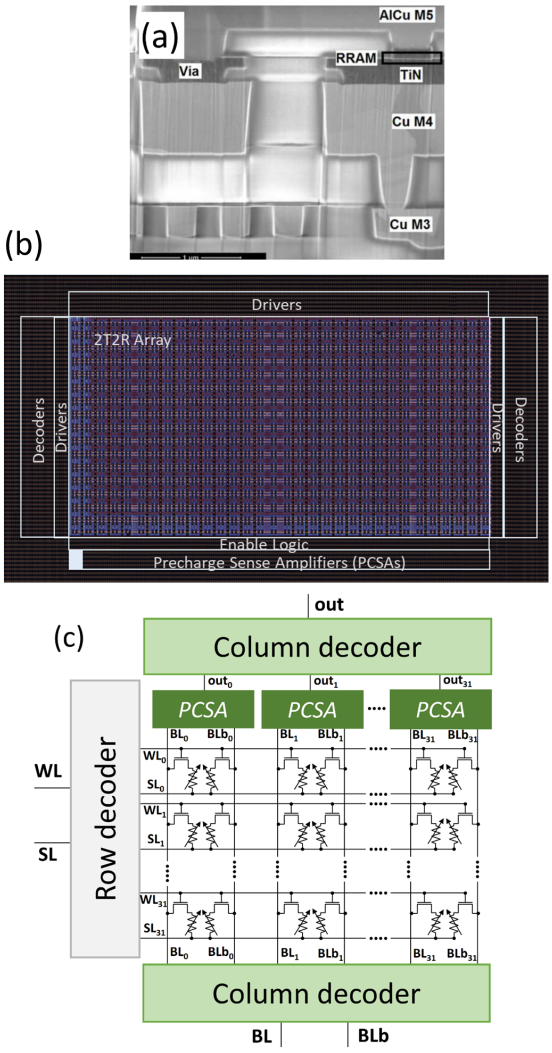

Fig. 1. (a) Electron microscopy image of a hafnium oxide resistive memory cell (RRAM) integrated in the backend-of-line of a 130nm CMOS process. (b) Photograph and (c) simplified schematic of the hybrid CMOS/RRAM test chip characterized in this work. The white rectangle in (b) materializes a single PCSA.

- We discuss the results, and compare our approach with the idea of storing three resistance levels per device.

Partial and preliminary results of this work have been presented at a conference [20]. This journal version adds the experimental characterization of bit errors in our architecture, supported by a comprehensive analysis of the impact on process, voltage, and temperature variations, and their impact at the neural network level, together with a detailed analysis of the use of ternary networks over binarized ones.

II. Background

The main equation in conventional neural networks is the computation of the neuronal activation $A_j = f\left(\sum_i W_{ji} X_i\right)$, where A_j, the synaptic weights W_{ji}, and input neuronal activations X_i assume real values, and f is a non-linear activation function. Binarized neural networks (BNNs) are a considerable simplification of conventional neural networks, in which all neuronal activations (A_j, X_i) and synaptic weights W_{ji} can only take binary values meaning $+1$ and -1. Neuronal activation then becomes:

$$A_j = \text{sign}\left(\sum_i XNOR\left(W_{ji}, X_i\right) - T_j\right), \qquad (1)$$

where sign is the sign function, T_j is a threshold associated with the neuron, and the $XNOR$ operation is defined in Table I. Training BNNs is a relatively sophisticated operation, during which each synapse needs to be associated with a real value in addition to its binary value (see Appendix). Once training is finished, these real values can be discarded, and the neural network is entirely binarized. Due to their reduced memory requirements, and reliance on simple arithmetic operations, BNNs are especially appropriate for in- or near- memory implementations. In particular, multiple groups investigate the implementation of BNN inference with resistive memory tightly integrated at the core of CMOS [6]–[13]. Usually, resistive memory stores the synaptic weights W_{ji}. However, this comes with a significant challenge: resistive memory is prone to bit errors, and in digital applications, is typically used with strong error-correcting codes (ECC). ECC, which requires large decoding circuits [21], goes against the principles of in- or near- memory computing. For this reason, [6] proposes a two-transistor/two-resistor (2T2R) structure, which reduces resistive memory bit errors, without the need for ECC decoding circuit, by storing synaptic weights in a differential fashion. This architecture allows the extremely efficient implementation of BNNs, and using the resistive memory devices in very favorable programming conditions (low energy, high endurance). It should be noted that systems using this architecture function with row-by-row read operations, and do not use the in-memory computing technique of using the Kirchhoff current law to perform the sum operation of neural networks, while reading all devices at the same time [22], [23]. This choice limits the parallelism of such architectures, while at the same time avoiding the need of analog-to-digital conversion and analog circuits such as operational amplifiers, as discussed in detail in [24].

In this work, we show that the same architecture can be used for a generalization of BNNs – ternary neural networks (TNNs)[1], where neuronal activations and synaptic weights A_j, X_i, and W_{ji} can now assume three values: $+1$, -1, and 0. Equation (1) now becomes:

$$A_j = \phi\left(\sum_i GXNOR\left(W_{ji}, X_i\right) - T_j\right). \qquad (2)$$

$GXNOR$ is the "gated" XNOR operation that realizes the product between numbers with values $+1$, -1 and 0 (Table I). ϕ is an activation function that outputs $+1$ if its input is greater than a threshold Δ, -1 if the input is lesser than $-\Delta$ and 0 otherwise. We show experimentally and by circuit simulation in sec. III how the 2T2R BNN architecture can be extended to TNNs with practically no overhead, in sec. V its bit errors, and in sec. VI the corresponding benefits in terms of neural network accuracy.

[1]In the literature, the name "Ternary Neural Networks" is sometimes also used to refer to neural networks where the synaptic weights are ternarized, but the neuronal activations remain real or integer [25], [26].

TABLE I
TRUTH TABLES OF THE XNOR AND GXNOR GATES

W_{ji}	X_i	$XNOR$
-1	-1	1
-1	1	-1
1	-1	-1
1	1	1

W_{ji}	X_i	$GXNOR$
-1	-1	1
-1	1	-1
1	-1	-1
1	1	1
0	X	0
X	0	0

III. THE OPERATION OF A PRECHARGE SENSE AMPLIFIER CAN PROVIDE TERNARY WEIGHTS

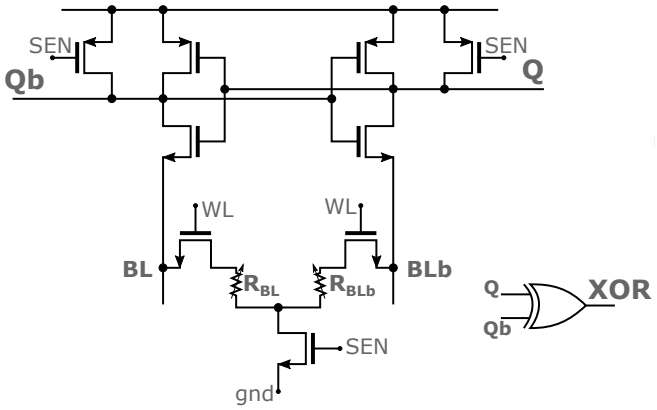

Fig. 2. Schematic of the precharge sense amplifier fabricated in the test chip.

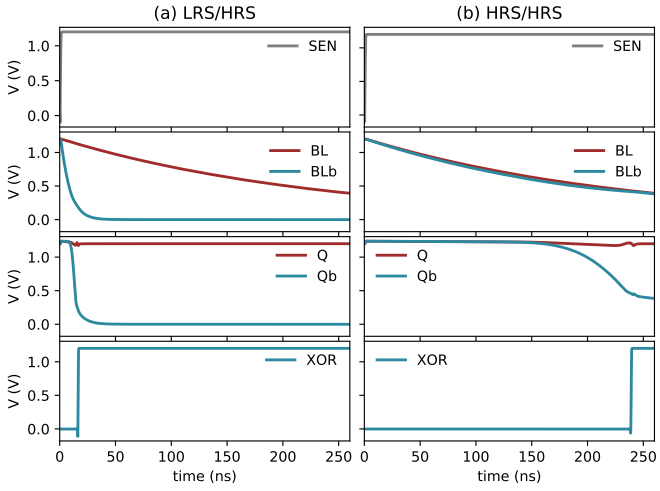

Fig. 3. Circuit simulation of the precharge sense amplifier of Fig. 2 with a supply voltage of 1.2V, using thick oxide transistors (nominal voltage of 5V) , if the two devices are programmed in an (a) LRS / HRS (5kΩ/350kΩ) or (b) HRS/HRS (320kΩ/350kΩ) configuration.

In this work, we use the architecture of [6], where synaptic weights are stored in a differential fashion. Each bit is implemented using two devices programmed either as low resistance state (LRS) / high resistance state (HRS) to mean weight $+1$ or HRS/LRS to mean weight -1. Fig. 1 presents the test chip used for the experiments. This chip cointegrates 130nm CMOS and resistive memory in the back-end-of-line, between levels four and five of metal. The resistive memory cells are based on 10nm thick hafnium oxide (Fig. 1(a)). All devices are integrated with a series NMOS transistor. After an initial forming step (consisting in the application of a voltage ramp from zero volts to 3.3V at a rate of $1000\text{V}/\text{s}$, and with a current limited to a compliance of $200\mu\text{A}$), the devices can switch between high resistance state (HRS) and low resistance state (LRS), through the dissolution or creation of conductive filaments of oxygen vacancies. Programming into the HRS is obtained by the application of a negative RESET voltage pulse (typically between 1.5V and 2.5V during $1\mu\text{s}$). Programming into the LRS is obtained by the application of a positive SET pulse (also typically between 1.5V and 2.5V during $1\mu\text{s}$), with current limited to a compliance current through the choice of the voltage applied on the transistor gate through the word line. This test chip is designed with highly conservative sizing, allowing the application of a wide range of voltages and electrical currents to the RRAM cells. The area of each bit cell is $6.6 \times 6.9\mu\text{m}^2$. More details on the RRAM technology are provided in [24].

Our experiments are based on a $2,048$ devices array incorporating all sense and periphery circuitry, illustrated in Fig. 1(b-c). The ternary synaptic weights are read using on-chip precharge sense amplifiers (PCSA), presented in Fig. 2, and initially proposed in [27] for reading spin-transfer magnetoresistive random access memory. Fig. 3(a) shows an electrical simulation of this circuit to explain its working principle, using the Mentor Graphics Eldo simulator. These first simulations are presented in the commercial 130nm ultra-low leakage technology, used in our test chip, with a low supply voltage of 1.2V [28], with thick oxide transistors (the nominal voltage in this process for thick oxide transistor is 5V). Since the technology targets ultra-low leakage applications the threshold voltages are significantly high (around 0.6V), thus a supply voltage of 1.2V significantly reduces the overdrive of the transistors ($V_{GS} - V_{TH}$).

In the first phase (SEN=0), the outputs Q and Qb are precharged to the supply voltage V_{DD}. In the second phase (SEN=V_{DD}), each branch starts to discharge to the ground. The branch that has the resistive memory (BL or BLb) with the lowest electrical resistance discharges faster and causes its associated inverter to drive the output of the other inverter to the supply voltage. At the end of the process, the two outputs are therefore complementary and can be used to tell which resistive memory has the highest resistance and therefore the synaptic weight. We observed that the convergence speed of a PCSA depends heavily on the resistance state of the two resistive memories. This effect is particularly magnified when the PCSA is used with a reduced overdrive, as presented here: the operation of the sense amplifier is slowed down, with regards to nominal voltage operation, and the convergence speed differences between resistance values become more apparent. Fig. 3(b) shows a simulation where the two devices, BL and BLb, were programmed in the HRS. We see that the two outputs converge to complementary values in more than 200ns, whereas less than 50ns were necessary in Fig. 3(a), where the

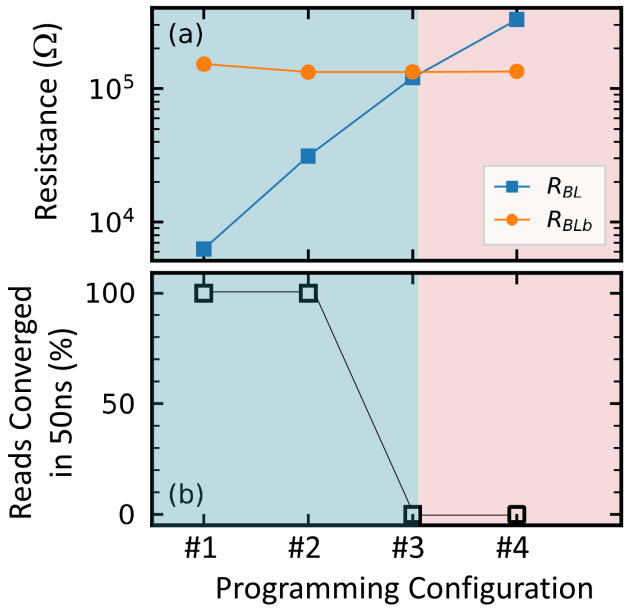

Fig. 4. Two devices have been programmed in four distinct programming conditions, presented in (a), and measured using an on-chip sense amplifier. (b) Proportion of read operations that have converged in 50ns, over 100 trials.

devices are programmed in complementary LRS/HRS states.

These first simulations suggest a technique for implementing ternary weights using the memory array of our test chip. Similarly to when this array is used to implement BNN, we propose to program the devices in the LRS/HRS configuration to mean the synaptic weight 1, and HRS/LRS to mean the synaptic weight -1. Additionally, we use the HRS/HRS configuration to mean synaptic weight 0, while the LRS/LRS configuration is avoided. The sense operation is performed during a duration of 50ns. If at the end of this period, outputs Q and Qb have differentiated, causing the output of the XOR gate to be 1, output Q determines the synaptic weight (1 or -1). Otherwise, the output of the XOR gate is 0, and the weight is determined to be 0.

This type of coding is reminiscent to the one used by the 2T2R ternary content-addressable memory (TCAM) cell of [29], where the LRS/HRS combination is used for coding 0, the HRS/LRS combination for coding 1, and the HRS/HRS combination for coding "don't care" (or X).

Experimental measurements on our test chip confirm that the PCSA can be used in this fashion. We first focus on one synapse of the memory array. We program one of the two devices (BLb) to a resistance of 100kΩ. We then program its complementary device BL to several resistance values, and for each of them perform 100 read operations of duration 50ns, using on-chip PCSAs.

These PCSAs are fabricated using thick-oxide transistors, designed for a nominal supply voltage of $5V$, and here used with a supply voltage of $1.2V$, close to their threshold voltage ($0.6V$), to reduce their overdrive, and thus to exacerbate the PCSA delay variations. In the test chip, they are sized conservatively with a total area of $290\mu m^2$. The use of thick oxide transistors in this test chip allows us to investigate the

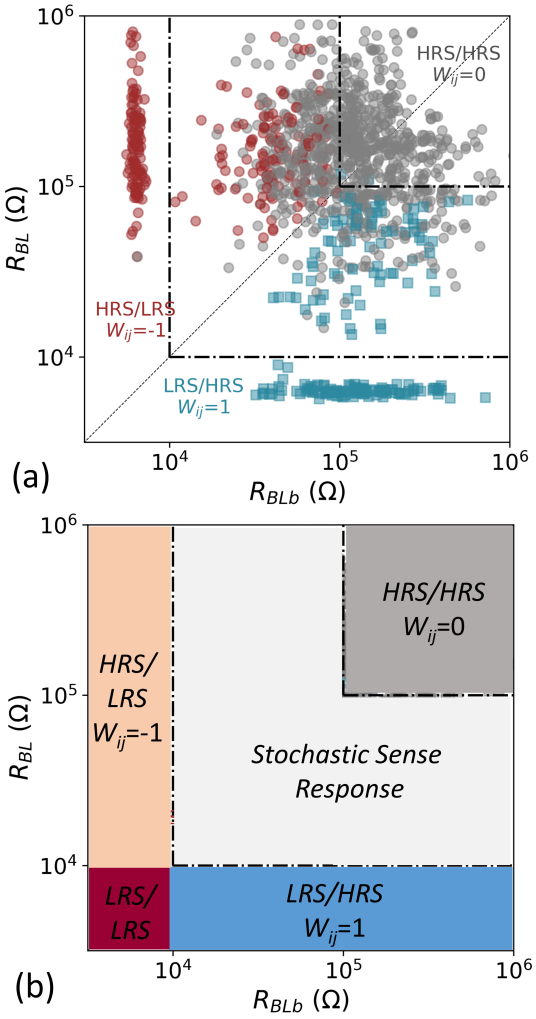

Fig. 5. For 109 device pairs programmed with multiple R_{BL}/R_{BLb} configuration, value of the synaptic weight measured by the on-chip sense amplifier using the strategy described in body text and 50ns reading time.

behavior of the devices at high voltages, without the concern of damaging the CMOS periphery circuits. Fig. 4 plots the probability that the sense amplifier has converged during the read time. In 50ns, the read operation is only converged if the resistance of the BL device is significantly lower than 100kΩ.

To evaluate this behavior in a broader range of programming conditions, we repeated the experiment on 109 devices and their complementary devices of the memory array programmed, every 14 times, with various resistance values in the resistive memory, and performed a read operation in 50ns with an on-chip PCSA. The memory array of our test chip features one separate PCSA per column. Therefore, 32 different PCSAs are used in our results. Fig. 5(a) shows, for each couple of resistance values R_{BL} and R_{BLb} if the read operation was converged with $Q = V_{DD}$ (blue), meaning a weight of 1, converged with $Q = 0$ (red), meaning a weight of -1, or not converged (grey) meaning a weight of 0.

The results confirm that LRS/HRS or HRS/LRS configurations may be used to mean weights 1 and -1, and HRS/HRS for weight 0. When both devices are in HRS (resistance higher

than $100\text{k}\Omega$, the PCSA never converges within 50ns (weight of 0). When one device is in LRS (resistance lower than $10\text{k}\Omega$, the PCSA always converges within 50ns (weight of ± 1). The separation between the 1 (or -1) and 0 regions is not strict, and for intermediate resistance values, we see that the read operation may or may not converge in 50ns. Fig. 5(b) summarizes the different operation regimes of the PCSA.

TABLE II
ERROR RATES ON TERNARY WEIGHTS MEASURED EXPERIMENTALLY

Programming Conditions	Type 1 ($1 \longleftrightarrow -1$)	Type 2 ($\pm 1 \rightarrow 0$)	Type 3 ($0 \rightarrow \pm 1$)
Fig. 7(a)	$< 10^{-6}$	$< 1\%$	6.5%
Fig. 7(b)	$< 10^{-6}$	$< 1\%$	18.5%

IV. IMPACT OF PROCESS, VOLTAGE, AND TEMPERATURE VARIATIONS

We now verify the robustness of the proposed scheme to process, voltage, and temperature variation. For this purpose, we performed extensive circuit simulations of the operation of the sense amplifier, reproducing the conditions of the experiments of Fig. 5, using the same resistance values for the RRAM devices, and including process, voltage, and temperature variations. The results of the simulations are processed and plotted using the same format as the experimental results of Fig. 5, to ease comparison.

These simulations are obtained using the Monte Carlo simulator provided by the Mentor Graphics Eldo tool with parameters validated on silicon, provided by the design kit of our commercial CMOS process. Each point in the graphs of Fig. 6 therefore features different transistor parameters. We included global and local process variations, as well as transistor mismatch, in order to capture the whole range of transistor variabilities observed in silicon. In order to assess the impact of voltage and temperature variations, these simulations are presented in three conditions: slow transistors ($0°\text{C}$ temperature, and 1.1V supply voltage, Fig. 6(a)), experimental conditions ($27°\text{C}$ temperature, and 1.2V supply voltage, Fig. 6(b)), and fast transistors ($60°\text{C}$ temperature, and 1.3V supply voltage, Fig. 6(c)). The RRAM devices are modeled by resistors. Their process variations are naturally included through the use of different resistance values in Fig. 6. The impact of voltage variation on RRAM is naturally included through Ohm's law, and the impact of temperature variation, which is smaller than on transistors, is neglected.

In all three conditions, the simulation results appear very similar to the experiments. Three clear regions are observed: non-convergence of the sense amplifier within $50ns$ for devices in HRS/HRS, and convergence within this time to a $+1$ or -1 value for devices in LRS/HRS and HRS/LRS, respectively. However, the frontier between these regimes is much sharper in the simulations than in the experiments. As the different data points in Fig. 6 differ by process and mismatch variations, this suggests that process variation does not cause the stochasticity observed in the experiments of Fig. 5, and that they have little impact in our scheme.

We also see that the frontier between the different sense regimes in all three operating conditions remains firmly within the $10 - 100k\Omega$ range, suggesting that even high variations of voltage ($\pm 0.1V$) and temperature ($\pm 30°C$) do not endanger the functionality of our scheme. Logically, in the case of fast transistors, the frontier is shifted toward higher resistances, whereas in the case of slow transistors, it is shifted toward lower resistances. Independent simulations allowed verifying that this change is mostly due to the voltage variations: the temperature variations have an almost negligible impact on the proposed scheme.

We also observed that the impact of voltage variations increased importantly when reducing the supply voltage. For example, with a supply voltage of $0.7V$ instead of the $1.2V$ value considered here, variations of the supply voltage of $\pm 0.1V$ can impact the mean switching delay of the PCSA, by a factor two. The thick oxide transistors used in this work have a nominal voltage of $5V$, and a typical threshold voltage of approximately $0.6V$. Therefore, although our scheme is especially appropriate for supply voltages far below the nominal voltage, it is not necessarily appropriate for voltages in the subthreshold regime, or very close to the threshold voltage.

V. PROGRAMMABILITY OF TERNARY WEIGHTS

To ensure reliable functioning of the ternary sense operation, we have seen that devices in LRS should be programmed to electrical resistance below $10\text{k}\Omega$, and devices in HRS to resistances above $100\text{k}\Omega$ (Fig. 5(b)). The electrical resistance of resistive memory devices depends considerably on their programming conditions [24], [30]. Fig. 7 shows the distributions of LRS and HRS resistances using two programming conditions, over the $2,048$ devices of the array, differentiating devices connected to bit lines and to bit lines bar. We see that in all cases, the LRS features a tight distribution. The SET process is indeed controlled by a compliance current that naturally stops the filament growth at a targeted resistance value [31]. An appropriate choice of the compliance current can ensure LRS below $10\text{k}\Omega$ in most situations.

On the other hand, the HRS shows a broad statistical distribution. In the RESET process, the filament indeed breaks in a random process, making it extremely hard to control the final state [31], [32]. The use of stronger programming conditions leads to higher values of the HRS.

This asymmetry between the variability of LRS and HRS means that in our scheme, the different ternary weight values feature different error rates naturally. The ternary error rates in the two programming conditions of Fig. 7(a) are listed in Table II. Errors of Type 1, where weight values of 1 and -1 are inverted are the least frequent. Errors of Type 2, where a weight value of 1 or -1 is replaced by a weight value of 0 are infrequent as well. On the other hand, due to the large variability of the HRS, weight values 0 have a significant probability to be measured as 1 or -1 (Type 3 errors): 6.5% in the conditions of Fig. 7(a), and 18.5% in the conditions of Fig. 7(b).

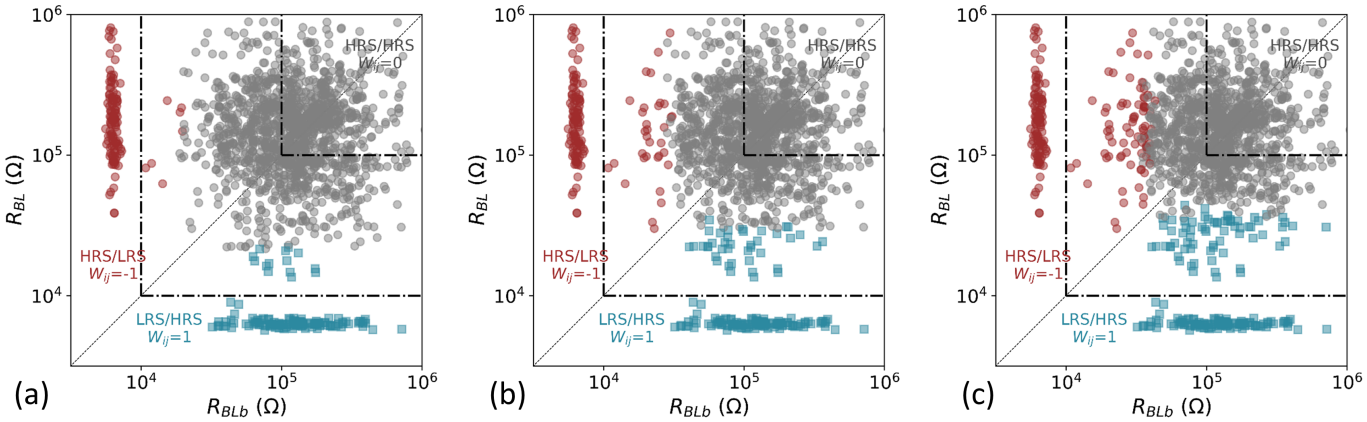

Fig. 6. Three Monte Carlo SPICE-based simulation of the experiments of Fig. 5, in three situations: (a) slow transistors (0°C temperature, 1.1V supply voltage), (b) experimental conditions (27°C temperature, 1.2V supply voltage), (c) fast transistors (60°C temperature, 1.3V supply voltage). The simulations include local and global process variations, as well as transistor mismatch, in a way that each point in the Figure is obtained using different transistor parameters. All results are plotted in the same manner and with the same conventions as Fig. 5.

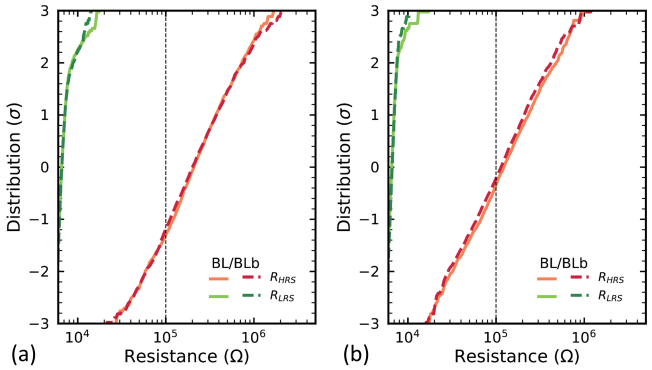

Fig. 7. Distribution of the LRS and HRS states programmed with a SET compliance of 200µA, RESET voltage of 2.5V and programming pulses of (a) 100µs and (b) 1µs. Measurements are performed 2,048 RRAM devices, separating bit line (full lines) and bit line bar (dashed lines) devices.

Some resistive memory technologies with large memory windows, such as specifically optimized conductive bridge memories [33], would feature lower Type 3 error rates. Similarly, program-and-verify strategies [34]–[36] may reduce this error rate. Nevertheless, the higher error rate for zeros than for 1 and -1 weights is an inherent feature of our architecture. Therefore, in the next section, we assess the impact of these errors on the accuracy of neural networks.

VI. Network-Level Implications

We first investigate the accuracy gain when using ternarized instead of binarized networks. We trained BNN and TNN versions of networks with Visual Geometry Group (VGG) type architectures [37] on the CIFAR-10 task of image recognition, consisting in classifying 1,024 pixels color images among ten classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck) [38]. Simulations are performed using PyTorch 1.1.0 [39] on a cluster of eight Nvidia GeForce RTX 2080 GPUs.

The architecture of our networks consists of six convolutional layers with kernel size three. The number of filters at the first layer is called N and is multiplied by two every two layers. Maximum-value pooling with kernel size two is used every two layers and batch-normalization [40] every layer. The classifier consists of one hidden layer of 512 units. For the TNN, the activation function has a threshold $\Delta = 5 \cdot 10^{-2}$ (as defined in section II). The training methods for both the BNN and the TNN are described in the Appendix. The training is performed using the AdamW optimizer [41], [42], with minibatch size 128. The initial learning rate is set to 0.01, and the learning rate schedule from [42], [43] (Cosine annealing with two restarts, for respectively 100, 200, 400 epochs) is used, resulting in a total of 700 epochs. Training data is augmented using random horizontal flip, and random choice between cropping after padding and random small rotations.

No error is added during the training procedure, as our device is meant to be used for inference. The synaptic weights encoded by device pairs would be set after the model has been trained on a computer.

Fig. 8 shows the maximum test accuracy resulting from these training simulations, for different sizes of the model. The error bars represent one standard deviation of the training accuracies. TNNs always outperform BNNs with the same model size (and, therefore, the same number of synapses). The most substantial difference is seen for smaller model size, but a significant gap remains even for large models. Besides, the difference in the number of parameters required to reach a given accuracy for TNNs and BNNs increases with higher accuracies. There is, therefore, a definite advantage to use TNNs instead of BNNs.

Fig. 8 compared fully ternarized (weights and activations) with regards to fully binarized (weights and activations) ones. Table III lists the impact of weight ternarization for different types of activations (binary, ternary, and real activation). All results are reported on a model of size $N = 128$, trained on CIFAR-10, and are averaged over five training procedures. We observe that for BNNs and TNNs with quantized acti-

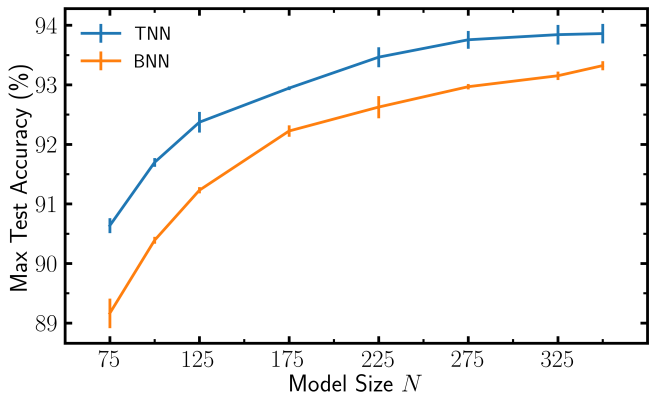

Fig. 8. Simulation of the maximum test accuracy reached during one training procedure, averaged over five trials, for BNNs and TNNs with various model sizes on the CIFAR-10 dataset. Error bar is one standard deviation.

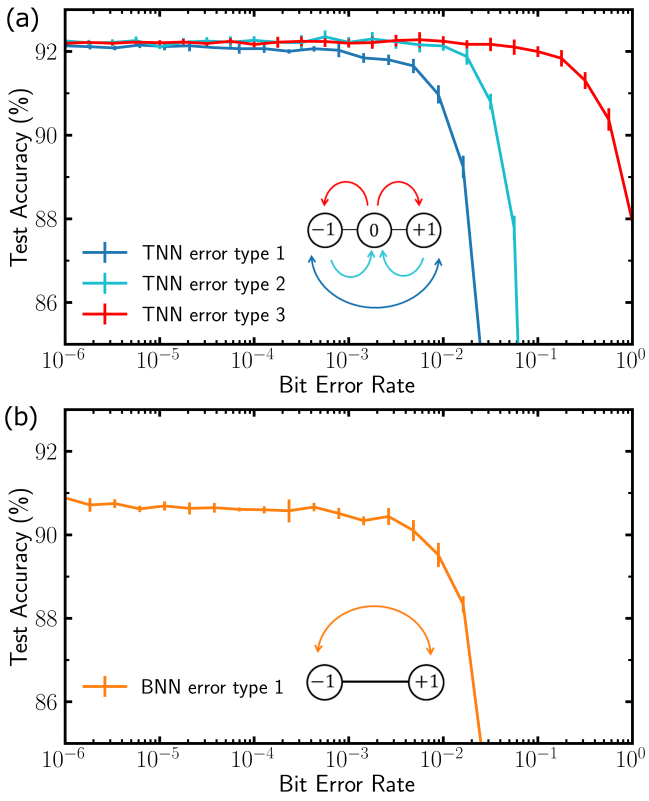

Fig. 9. Simulation of the impact of Bit Error Rate on the test accuracy at inference time for model size $N = 128$ TNN in (a) and BNN in (b). Type 1 errors are sign switches (e.g. $+1$ mistaken for -1), Type 2 errors are ± 1 mistaken for 0, and Type 3 errors are 0 mistaken for ± 1, as described in the inset schematics. Errors are sampled at each mini batch and the test accuracy is averaged over five passes through the test set. Error bars are one standard deviation. The bit error rate is given as an absolute rate.

vations, the accuracy gains provided by ternary weights over binary weights are 0.84 and 0.86 points and are statistically significant over the standard deviations. This accuracy gain is more important than the gain provided by ternary activations over binary activations, which is about 0.3 points. This bigger impact of weight ternarization over ternary activation may come from the ternary kernels having a better expressing power over binary kernels, which are often redundant in practical settings [3]. The gain of ternary weights drops to 0.26 points if real activation is allowed (using rectified linear unit, or ReLU, as activation function, see appendix), and is not statistically significant considering the standard deviations.

Quantized activations are vastly more favorable in the context of hardware implementations, and in this situation, there is thus a statistically significant benefit provided by ternary weights over binary weights.

TABLE III
COMPARISON OF THE GAIN IN TEST ACCURACY FOR A $N = 128$ MODEL SIZE ON CIFAR-10 OBTAINED BY WEIGHT TERNARIZATION INSTEAD OF BINARIZATION FOR THREE TYPES OF ACTIVATION QUANTIZATION.

	Activations		
	Binary	Ternary	Full Precision
Weights			
Binary	91.19 ± 0.08	91.51 ± 0.09	93.87 ± 0.19
Ternary	92.03 ± 0.12	92.35 ± 0.05	94.13 ± 0.10
Gain of ternarization	*0.84*	*0.86*	*0.26*

We finally investigate the impact of bit errors in BNNs and TNNs to see if the advantage provided by using TNNs in our approach remains constant when errors are taken into account. Consistently with the results reported in section V, three types of errors are investigated: Type 1 errors are sign switches, e.g., $+1$ mistaken for -1, Type 2 errors are only defined for TNNs and correspond to ± 1 mistaken for 0, and Type 3 errors are 0 mistaken for ± 1, as illustrated in the inset schematic of Fig. 9(a).

Fig. 9(a) shows the impact of these errors on the test accuracy for different values of the error rate at inference time. These simulation results are presented on CIFAR-10 with a model size of $N = 128$. Errors are randomly and artificially introduced in the weights of the neural network . Bit errors are included at the layer level and sampled at each mini-batch of the test set. Type 1 errors switch the sign of a synaptic weight with a probability equal to the rate of type 1 errors. Type 2 errors set a non-zero synaptic weight to 0 with a probability equal to the type 2 error rate. Type 3 errors set a synaptic weight of 0 to ± 1 with a probability equal to the type 3 error rate, the choice of the sign ($+1$ or -1) is made with 0.5 probability. Fig. 9 is obtained by averaging the test accuracy obtained for five passes through the test set for increasing bit error rate.

Type 1 errors have the most impact on neural network accuracy. As seen in Fig. 9(b), the impact of these errors is similar to the impact of weight errors in a BNN. On the other hand, Type 3 errors have the least impact, with bit error rates

as high as 20% degrading surprisingly little the accuracy. This result is fortunate, as we have seen in section V that Type 3 errors are the most frequent in our architecture.

We also performed simulations considering all three types of error at the same time, with error rates reported in Table II corresponding to the programming conditions of Fig. 7(a) and 7(b). For Type 1 and Type 2 errors, we considered the upper limits listed in Table II. For the conditions of Fig. 7(a) (Type 3 error rate of 6.5%), the test accuracy was degraded from 92.2% to $92.05 \pm 0.14\%$, and to $92.02 \pm 0.17\%$ for the conditions of 7(b) (Type 3 error rate of 18.5%), where the average and standard deviation is performed over 100 passes through the test set. We found that the slight degradation on CIFAR-10 test accuracy was mostly due to the Type 2 errors, although Type 3 errors are much more frequent.

The fact that mistaking a 0 weight for a ± 1 weight (Type 3 error) has much less impact than mistaking a ± 1 weight for a 0 weight (Type 2 error) can seem surprising. However, it is known, theoretically and practically, that in BNNs, some weights have little importance to the accuracy of the neural networks [44]. They typically correspond to synapses that feature a 0 weight in a TNN, whereas synapses with ± 1 weights in a TNN correspond to "important" synapses of a BNN. It is thus understandable that errors on such synapses have more impact on the final accuracy of the neural network.

VII. Comparison with Three-Level Programming

An alternative approach to implementing ternary weights with resistive memory can be to program the individual devices into there separate levels. This idea is feasible, as the resistance level of the LRS can to a large extent be controlled through the choice of the compliance current during the SET operation in many resistive memory technologies [24], [31].

The obvious advantage of this approach is that it requires a single device per synapse. This idea also brings several challenges. First, the sense operation has to be more complex. The most natural technique is to perform two sense operations, comparing the resistance of a device under test to two different thresholds. Second, this technique is much more prone to bit errors than our technique, as states are not programmed in a differential fashion [24]. Additionally, this approach does not feature the natural resilience to Type 1 and Type 2 errors, and Type 2 and Type 3 errors will typically feature similar rates. Finally, unlike ours, this approach is prone to resistive drift, inherent to some resistive memory technologies [45].

These comments suggest that the choice of a technique for storing ternary weights should be dictated by technology. Our technique is especially appropriate for resistive memories not supporting single-device multilevel storage, with high error rates, or resistance drift. The three-levels per devices approach would be the most appropriate with devices with well controlled analog storage properties.

VIII. Conclusion

In this work, we revisited a differential memory architecture for BNNs. We showed experimentally on a hybrid CMOS/RRAM chip that, its sense amplifier can differentiate not only the LRS/HRS and HRS/LRS states, but also the HRS/HRS states in a single sense operation. This feature allows the architecture to store ternary weights, and to provide a building block for ternary neural networks. We showed by neural network simulation on the CIFAR-10 task the benefits of using ternary instead of binary networks, and the high resilience of TNNs to weights errors, as the type of errors observed experimentally in our scheme is also the type of errors to which TNNs are the most immune. This resilience allows the use of our architecture without relying on any formal error correction. Our approach also appears resilient to process, voltage, and temperature variation if the supply voltage remains reasonably higher than the threshold voltage of the transistors.

As this behavior of the sense amplifier is exacerbated at supply voltages below the nominal voltage, our approach especially targets extremely energy-conscious applications such as uses within wireless sensors or medical applications. This work opens the way for increasing the edge intelligence in such contexts, and also highlights that the low voltage operation of circuits may sometimes provide opportunities for new functionalities.

Acknowledgments

The authors would like to thank M. Ernoult and L. Herrera Diez for fruitful discussions.

Appendix: Training Algorithm of Binarized and Ternary Neural Networks

During the training of BNNs and TNNs, each quantized (binary or ternary) weight is associated with a real hidden weight. This approach to training quantized neural network was introduced in [3] and is presented in Algorithm 1.

The quantized weights are used for computing neuron values (equations (1) and (2)), as well as the gradients values in the backward pass. However, training steps are achieved by updating the real hidden weights. The quantized weight is then determined by applying to the real value the quantizing function $\mathrm{Quantize}$, which is ϕ for ternary or sign for binary as defined in section II. The quantization of activations is done by applying the same function $\mathrm{Quantize}$, except for real activation, which is done by applying a rectified linear unit ($\mathrm{ReLU}(x) = \max(0, x)$).

Quantized activation functions (ϕ or sign) have zero derivatives almost everywhere, which is an issue for backpropagating the error gradients through the network. A way around this issue is the use of a straight-through estimator [46], which consists in taking the derivative of another function instead of the almost everywhere zero derivatives. Throughout this work, we take the derivative of $\mathrm{Hardtanh}$, which is 1 between -1 and 1 and 0 elsewhere, both for binary and ternary activations.

The simulation code used in this work is available publicly in the Github repository: https://github.com/Laborieux-Axel/Quantized_VGG

Algorithm 1 Training procedure for binary and ternary neural networks. W^{h} are the hidden weights, $\theta^{\mathrm{BN}} = (\gamma_l, \beta_l)$ are Batch Normalization parameters, U_W and U_θ are the parameter updates prescribed by the Adam algorithm [41], (X, y) is a batch of labelled training data, and η is the learning rate. "cache" denotes all the intermediate layers computations needed to be stored for the backward pass. $\mathrm{Quantize}$ is either ϕ or sign as defined in section II. " $\cdot$ " denotes the element-wise product of two tensors with compatible shapes.

Input: W^{h}, $\theta^{\mathrm{BN}} = (\gamma_l, \beta_l)$, U_W, U_θ, (X, y), η.
Output: W^{h}, θ^{BN}, U_W, U_θ.

$W^{\mathrm{Q}} \leftarrow \mathrm{Quantize}(W^{\mathrm{h}})$ ▷ Computing quantized weights
$A_0 \leftarrow X$ ▷ Input is not quantized
for $l = 1$ to L **do** ▷ For loop over the layers
  $z_l \leftarrow W_l^{\mathrm{Q}} A_l$ ▷ Matrix multiplication
  $A_l \leftarrow \gamma_l \cdot \frac{z_l - \mathrm{E}(z_l)}{\sqrt{\mathrm{Var}(z_l) + \epsilon}} + \beta_l$ ▷ Batch Normalization [40]
  if $l < L$ **then** ▷ If not the last layer
    $A_l \leftarrow \mathrm{Quantize}(A_l)$ ▷ Activation is quantized
  end if
end for
$\hat{y} \leftarrow A_L$
$C \leftarrow \mathrm{Cost}(\hat{y}, y)$ ▷ Compute mean loss over the batch
$(\partial_W C, \partial_\theta C) \leftarrow \mathrm{Backward}(C, \hat{y}, W^{\mathrm{Q}}, \theta^{\mathrm{BN}}, \mathrm{cache})$ ▷ Cost gradients
$(U_W, U_\theta) \leftarrow \mathrm{Adam}(\partial_W C, \partial_\theta C, U_W, U_\theta)$
$W^{\mathrm{h}} \leftarrow W^{\mathrm{h}} - \eta U_W$
$\theta^{\mathrm{BN}} \leftarrow \theta^{\mathrm{BN}} - \eta U_\theta$
return W^{h}, θ^{BN}, U_W, U_θ

REFERENCES

[1] X. Xu, Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu, and Y. Shi, "Scaling for edge inference of deep neural networks," *Nature Electronics*, vol. 1, no. 4, p. 216, 2018.

[2] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6869–6898, 2017.

[3] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016.

[4] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *Proc. ECCV*. Springer, 2016, pp. 525–542.

[5] X. Lin, C. Zhao, and W. Pan, "Towards accurate binary convolutional neural network," in *Advances in Neural Information Processing Systems*, 2017, pp. 345–353.

[6] M. Bocquet, T. Hirztlin, J.-O. Klein, E. Nowak, E. Vianello, J.-M. Portal, and D. Querlioz, "In-memory and error-immune differential rram implementation of binarized deep neural networks," in *IEDM Tech. Dig.* IEEE, 2018, p. 20.6.1.

[7] S. Yu, Z. Li, P.-Y. Chen, H. Wu, B. Gao, D. Wang, W. Wu, and H. Qian, "Binary neural network with 16 mb rram macro chip for classification and online training," in *IEDM Tech. Dig.* IEEE, 2016, pp. 16–2.

[8] E. Giacomin, T. Greenberg-Toledo, S. Kvatinsky, and P.-E. Gaillardon, "A robust digital rram-based convolutional block for low-power image processing and learning applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 2, pp. 643–654, 2019.

[9] X. Sun, S. Yin, X. Peng, R. Liu, J.-s. Seo, and S. Yu, "Xnor-rram: A scalable and parallel resistive synaptic architecture for binary neural networks," *algorithms*, vol. 2, p. 3, 2018.

[10] Z. Zhou, P. Huang, Y. Xiang, W. Shen, Y. Zhao, Y. Feng, B. Gao, H. Wu, H. Qian, L. Liu *et al.*, "A new hardware implementation approach of bnns based on nonlinear 2t2r synaptic cell," in *2018 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2018, pp. 20–7.

[11] M. Natsui, T. Chiba, and T. Hanyu, "Design of mtj-based nonvolatile logic gates for quantized neural networks," *Microelectronics journal*, vol. 82, pp. 13–21, 2018.

[12] T. Tang, L. Xia, B. Li, Y. Wang, and H. Yang, "Binary convolutional neural network on rram," in *Proc. ASP-DAC*. IEEE, 2017, pp. 782–787.

[13] J. Lee, J. K. Eshraghian, K. Cho, and K. Eshraghian, "Adaptive precision cnn accelerator using radix-x parallel connected memristor crossbars," *arXiv preprint arXiv:1906.09395*, 2019.

[14] H. Alemdar, V. Leroy, A. Prost-Boucle, and F. Pétrot, "Ternary neural networks for resource-efficient ai applications," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 2547–2554.

[15] L. Deng, P. Jiao, J. Pei, Z. Wu, and G. Li, "Gxnor-net: Training deep neural networks with ternary weights and activations without full-precision memory under a unified discretization framework," *Neural Networks*, vol. 100, pp. 49–58, 2018.

[16] K. Ando, K. Ueyoshi, K. Orimo, H. Yonekawa, S. Sato, H. Nakahara, M. Ikebe, T. Asai, S. Takamaeda-Yamazaki, T. Kuroda *et al.*, "Brein memory: A 13-layer 4.2 k neuron/0.8 m synapse binary/ternary reconfigurable in-memory deep neural network accelerator in 65 nm cmos," in *Proc. VLSI Symp. on Circuits*. IEEE, 2017, pp. C24–C25.

[17] A. Prost-Boucle, A. Bourge, F. Pétrot, H. Alemdar, N. Caldwell, and V. Leroy, "Scalable high-performance architecture for convolutional ternary neural networks on fpga," in *2017 27th International Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 2017, pp. 1–7.

[18] Z. Li, P.-Y. Chen, H. Xu, and S. Yu, "Design of ternary neural network with 3-d vertical rram array," *IEEE Transactions on Electron Devices*, vol. 64, no. 6, pp. 2721–2727, 2017.

[19] B. Pan, D. Zhang, X. Zhang, H. Wang, J. Bai, J. Yang, Y. Zhang, W. Kang, and W. Zhao, "Skyrmion-induced memristive magnetic tunnel junction for ternary neural network," *IEEE Journal of the Electron Devices Society*, vol. 7, pp. 529–533, 2019.

[20] A. Laborieux, M. Bocquet, T. Hirtzlin, J.-O. Klein, L. H. Diez, E. Nowak, E. Vianello, J.-M. Portal, and D. Querlioz, "Low power in-memory implementation of ternary neural networks with resistive ram-based synapse," in *2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2020.

[21] S. Gregori, A. Cabrini, O. Khouri, and G. Torelli, "On-chip error correcting techniques for new-generation flash memories," *Proc. IEEE*, vol. 91, no. 4, pp. 602–616, 2003.

[22] M. Prezioso *et al.*, "Training and operation of an integrated neuromorphic network based on metal-oxide memristors," *Nature*, vol. 521, no. 7550, p. 61, 2015.

[23] S. Ambrogio *et al.*, "Equivalent-accuracy accelerated neural-network training using analogue memory," *Nature*, vol. 558, p. 60, 2018.

[24] T. Hirtzlin, M. Bocquet, B. Penkovsky, J.-O. Klein, E. Nowak, E. Vianello, J.-M. Portal, and D. Querlioz, "Digital biologically plausible implementation of binarized neural networks with differential hafnium oxide resistive memory arrays," *Frontiers in Neuroscience*, vol. 13, p. 1383, 2020.

[25] N. Mellempudi, A. Kundu, D. Mudigere, D. Das, B. Kaul, and P. Dubey, "Ternary neural networks with fine-grained quantization," *arXiv preprint arXiv:1705.01462*, 2017.

[26] E. Nurvitadhi, G. Venkatesh, J. Sim, D. Marr, R. Huang, J. Ong Gee Hock, Y. T. Liew, K. Srivatsan, D. Moss, S. Subhaschandra *et al.*, "Can fpgas beat gpus in accelerating next-generation deep neural networks?" in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*. ACM, 2017, pp. 5–14.

[27] W. Zhao, C. Chappert, V. Javerliac, and J.-P. Noziere, "High speed, high stability and low power sensing amplifier for mtj/cmos hybrid logic circuits," *IEEE Transactions on Magnetics*, vol. 45, no. 10, pp. 3784–3787, 2009.

[28] R. G. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge, "Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits," *Proc. IEEE*, vol. 98, no. 2, pp. 253–266, Feb 2010.

[29] R. Yang, H. Li, K. K. Smithe, T. R. Kim, K. Okabe, E. Pop, J. A. Fan, and H.-S. P. Wong, "Ternary content-addressable memory with mos 2 transistors for massively parallel data search," *Nature Electronics*, vol. 2, no. 3, pp. 108–114, 2019.

[30] A. Grossi, E. Nowak, C. Zambelli, C. Pellissier, S. Bernasconi, G. Cibrario, K. El Hajjam, R. Crochemore, J. Nodin, P. Olivo *et al.*, "Fundamental variability limits of filament-based rram," in *IEDM Tech. Dig.* IEEE, 2016, pp. 4–7.

[31] M. Bocquet, D. Deleruyelle, H. Aziza, C. Muller, J.-M. Portal, T. Cabout, and E. Jalaguier, "Robust compact model for bipolar oxide-based resistive switching memories," *IEEE transactions on electron devices*, vol. 61, no. 3, pp. 674–681, 2014.

[32] D. R. B. Ly *et al.*, "Role of synaptic variability in resistive memory-based spiking neural networks with unsupervised learning," *J. Phys. D: Applied Physics*, 2018.

[33] E. Vianello, O. Thomas, G. Molas, O. Turkyilmaz, N. Jovanović, D. Garbin, G. Palma, M. Alayan, C. Nguyen, J. Coignus *et al.*, "Resistive memories for ultra-low-power embedded computing design," in *2014 IEEE International Electron Devices Meeting*. IEEE, 2014, pp. 6–3.

[34] S. R. Lee, Y.-B. Kim, M. Chang, K. M. Kim, C. B. Lee, J. H. Hur, G.-S. Park, D. Lee, M.-J. Lee, C. J. Kim *et al.*, "Multi-level switching of triple-layered taox rram with excellent reliability for storage class memory," in *2012 Symposium on VLSI Technology (VLSIT)*. IEEE, 2012, pp. 71–72.

[35] F. Alibart, L. Gao, B. D. Hoskins, and D. B. Strukov, "High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," *Nanotechnology*, vol. 23, no. 7, p. 075201, 2012.

[36] C. Xu, D. Niu, N. Muralimanohar, N. P. Jouppi, and Y. Xie, "Understanding the trade-offs in multi-level cell reram memory design," in *2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2013, pp. 1–6.

[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[38] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[39] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[42] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in adam," *arXiv preprint arXiv:1711.05101*, 2017.

[43] ——, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[44] A. Laborieux, M. Ernoult, T. Hirtzlin, and D. Querlioz, "Synaptic metaplasticity in binarized neural networks," *arXiv preprint arXiv:2003.03533*, 2020.

[45] J. Li, B. Luan, and C. Lam, "Resistance drift in phase change memory," in *2012 IEEE International Reliability Physics Symposium (IRPS)*. IEEE, 2012, pp. 6C–1.

[46] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

Axel Laborieux Axel Laborieux received the M.S. degree in condensed matter physics from the Université Paris-Saclay, France, in 2018, where he is currently pursuing the Ph.D. degree in neuromorphic computing. His research interest includes the benefits brought by complex synapse behaviors in binarized neural networks and their physical implementation using spintronic nanodevices.

Marc Bocquet Marc Bocquet received the M.S. degree in electrical engineering and the Ph.D. degree in electrical engineering from the University of Grenoble, France, in 2006 and 2009, respectively. He is currently an Associate Professor with the Institute of Materials, Microelectronics, and Nanosciences of Provence, IM2NP, Université of Aix-Marseille and Toulon. His research interests include memory model, memory design, characterization, and reliability.

Tifenn Hirtzlin Tifenn Hirtzlin received the M.S. degree in nanosciences and electronics from the Université Paris-Sud, France, in 2017, where he is currently pursuing the Ph.D. degree in electrical engineering. His research interest includes designing intelligent memory-chip for low energy hardware data processing using bio-inspired concepts as a probabilistic approach to brain function and more conventional neural network approaches.

Jacques-Olivier Klein Jacques-Olivier Klein (M'90) received the Ph.D. degree from the Université Paris-Sud, France, in 1995, where he is currently a Full Professor. He focuses on the architecture of circuits and systems based on emerging nanodevices in the field of nanomagnetism and bio-inspired nanoelectronics. He is also a Lecturer with the Institut Universitaire de Technologie (IUT), Cachan. He has authored more than 100 technical papers.

Etienne Nowak Etienne Nowak received the M.Sc. degree in microelectronics from Grenoble University, Grenoble, France; Polito di Torino, Turin, Italy; and the Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2007, and the Ph.D. degree from the Institut National Polytechnique de Grenoble, Grenoble, France, in 2010. From 2010 to 2014, he was a Senior Engineer at the Semiconductor Research and Development Center, Samsung Electronics, Hwaseong, South Korea, where he was involved in the first generations of vertical nand flash memory. He joined CEA-Leti, Grenoble, France, in 2014, as a Project Manager on emerging nonvolatile memory. He published over 30 papers and holds two patents on these topics. Since 2017, he has been appointed as the Head of the Advanced Memory Device Laboratory, CEA-Leti, Grenoble, France, dedicated to nonvolatile memory backend technologies.

Elisa Vianello Elisa Vianello received the Ph.D. degree in microelectronics from the University of Udine, Udine, Italy, and the Polytechnic Institute of Grenoble, Grenoble, France, in 2009. She has been a Scientist with the Laboratoire d'Electronique des Technologies de l'Information, Commissariat à l'Energie Atomique et aux Energies Alternatives, Grenoble, since 2011. Her current research interests include resistive switching memory devices and selectors and the use of nanotechnologies for memory-centric computing and neuromorphic systems.

Jean-Michel Portal Jean-Michel Portal graduated in Electronic Engineering in 1996 and received the Ph.D. degree in Computer Sciences in 1999. is currently a Full professor in Electronics at Aix-Marseille University, where he heads the electronic department of the Institute of Materials, Microelectronics, and Nanosciences of Provence (IM2NP). His research interests include emerging non-volatile memory design and neuromorphic applications. He is author or co-author of more than 200 articles in International Refereed Journals and Conferences, and is a co-inventor of six patents. He has supervised 20 Ph.D. students. He is a recipient of the NanoArch 2012, Newcas 2013 and IEEE Transactions on Circuits and Systems Guillemin-Cauer 2017 Best Paper Awards.

Damien Querlioz Damien Querlioz (M'09) received the predoctoral education from the Ecole Normale Supérieure, Paris, and the Ph.D. degree from Université Paris-Sud, in 2008. After postdoctoral appointments at Stanford University and CEA, he became a Permanent Researcher with the Centre for Nanoscience and Nanotechnology of CNRS and Université Paris-Saclay. He focuses on novel usages of emerging non-volatile memory, in particular relying on inspirations from biology and machine learning. He coordinates the INTEGNANO Interdisciplinary Research Group. In 2016, he was a recipient of the European Research Council Starting Grant to develop the concept of natively intelligent memory. In 2017, he received the CNRS Bronze Medal.