Binarized SNNs: Efficient and Error-Resilient Spiking Neural Networks through Binarization

Ming-Liang Wei, Mikail Yayla, Shu-Yin Ho, Jian-Jia Chen, Chia-Lin Yang, Hussam Amrouch

TU Dortmund, Department of Computer Science, Dortmund, Germany

Citation: https://doi.org/10.1109/ICCAD51958.2021.9643463

BibTeX:

```
@inproceedings{9643463,
  author={Wei, Ming-Liang and Yayla, Mikail and Ho, Shu-Yin and Chen, Jiap-Jia and Yang, Chia-Lin and Am
  booktitle={2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)},
  title={Binarized SNNs: Efficient and Error-Resilient Spiking Neural Networks through Binarization},
  year={2021},
  volume={},
  number={},
  pages={1-9},
  doi={10.1109/ICCAD51958.2021.9643463}}
```

Binarized SNNs: Efficient and Error-Resilient Spiking Neural Networks through Binarization

Ming-Liang Wei*†, Mikail Yayla‡, Shu-Yin Ho*†, Jian-Jia Chen‡, Chia-Lin Yang*, Hussam Amrouch§
*National Taiwan University, †Macronix International Co., Ltd., ‡Technical University of Dortmund, §University of Stuttgart
Corresponding author e-mails: d04943004@ntu.edu.tw, mikail.yayla@tu-dortmund.de, amrouch@iti.uni-stuttgart.de

Preprint Version. Citation Info: 10.1109/ICCAD51958.2021.9643463

***Abstract*—Spiking Neural Networks (SNNs) are considered the third generation of NNs and can reach similar accuracy as conventional deep NNs, but with a considerable improvement in efficiency. However, to achieve high accuracy, state-of-the-art SNNs employ stochastic spike coding of the inputs, requiring multiple cycles of computation. Because of this and due to the nature of analog computing, it is required to accumulate and hold the charges of multiple cycles, necessitating a large membrane capacitor. This results in high energy, long latency, and expensive area costs, constituting one of the major bottlenecks in analog SNN implementations. Membrane capacitor size determines the precision of the firing time. Hence reducing the capacitor size considerably degrades the inference accuracy. To alleviate this, we focus on bridging the gap between binarized NNs (BNNs) and SNNs. BNNs are rapidly emerging as an attractive alternative for NNs due to their high efficiency and error tolerance. In this work, we evaluate the impact of deploying error-resilient BNNs, i.e. BNNs that have been proactively trained in the presence of errors, on analog implementation of SNNs. We show that for BNNs, the capacitor size and latency can be reduced significantly compared to state-of-the-art SNNs, which employ multi-bit models. Our experiments demonstrate that when error-resilient BNNs are deployed on analog-based SNN accelerator, the size of the membrane capacitor is reduced by 50%, the inference latency is decreased by two orders of magnitude, and energy is reduced by 57% compared to the baseline 4-bit SNN implementation, under minimal accuracy cost.**

I. Introduction

The success of deep neural networks (DNNs) has brought significant benefits to numerous fields while impacting our daily lives. However, the high inference accuracy comes at the cost of large resource demand, because NNs require a huge number of parameters and a massive number of multiply-accumulate (MAC) operations. This poses an immense challenge, because high-performing NN models are becoming increasingly larger, while low-power operation for sustainability is rapidly gaining importance in a wide range of application domains, especially for embedded systems.

Spiking Neural Networks (SNNs): SNNs is a computing scheme that is heavily influenced by the dynamics of biological neurons and are similar to them. In SNNs, neural activity is event-driven and described by the integration of voltage spikes over time. When a neuron receives a certain number of spikes, a predetermined threshold potential may be passed, causing the *firing* of an output spike. The sparse-spike-based operations use efficient coding and enable low-power operation, because accelerators built with simple analog or digital components can be employed for computation [1].

Existing challenges in SNNs: One major challenge in SNNs is to efficiently processe inputs coded in the time domain without influence on inference accuracy. Although recent studies claim that temporal input coding with one bit can implement multi-bit multiplication through single bit operations, they are not yet proven to be effective on complex tasks [2], [3], [4]. If, instead of temporal coding, stochastic encoding of the inputs is used, the high accuracy from well-trained models are inherited. However, stochastic coding necessitates many computation cycles to achieve high accuracy. Due to this, in analog computing based implementations, a large membrane capacitor (C_{mem}) is inevitably required to reliably accumulate and hold the charges of multiple cycles. Otherwise, the required inference accuracy cannot be sustained. A large C_{mem} size results in a high energy, long latency, and expensive area cost, constituting one of the major bottlenecks in SNN hardware accelerators [5], [6], [7]. Although reducing the size of C_{mem} has numerous benefits, it, rapidly degrades the inference accuracy because it impacts the precision of the firing time, which encodes the inner product result. Hence, if the SNN is error-prone, reducing C_{mem} size will aggressively degrade the inference results. *All in all, efficient and error-resilient SNN models allow the optimization of membrane capacitors resulting in energy, area, and speed improvements with minimal accuracy loss. Achieving this goal is concisely the focus of this work.*

Binarized Neural Networks (BNNs): Due to the light-weight implementation, BNNs have recently received remarkable attention since their inception [8], [9]. The binarization has three major benefits on the efficiency of NNs. (1) BNNs have significantly smaller model size and hence reduce movements of data between memories and computing units. (2) Binarization of the weights and activations enables replacing complex MAC circuits with simple XNOR logic gates. (3) BNNs can be optimized to achieve high error resiliency [10], [11].

Binarized Spiking Neural Networks (BSNNs): To optimize SNN computations by C_{mem} size reduction while sustaining high inference accuracy, we investigate the effectiveness of SNNs based on binary representation of model parameters. SNNs and BNNs synergise outstandingly in their motivations, and were designed with similar objectives in mind. For both BNNs and SNNs, energy efficiency, low latency, and error tolerance are crucial properties. Therefore, recent studies have focused on deploying BNNs on SNN

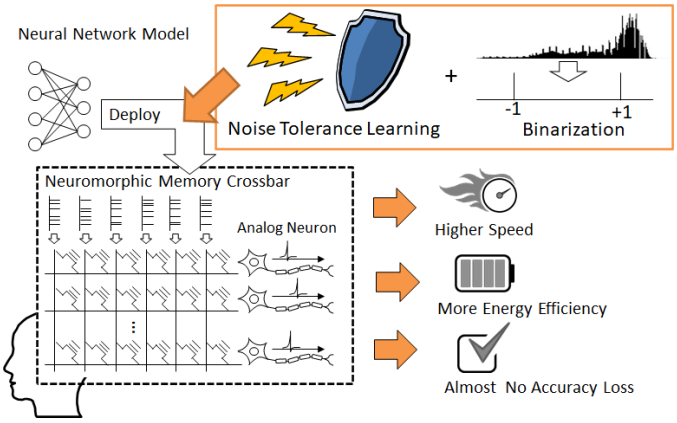


Fig. 1: Deploying binarized and error-resilient models on neuromorphic memory crossbar increases speed and energy efficiency with minimal accuracy cost.

hardware [12], [13]. The promise of BNNs for SNNs is twofold: First, concerning computations, XNOR can be used instead of multiplications, and at the same time, *stochastic input representation is not anymore necessary* leading to single-sample computation. Secondly, BNNs can be optimized for high error resiliency, which enables significant reduction of C_{mem} size. This leads to a profound increase in the efficiency of SNNs because C_{mem} size determines energy, latency, and chip area. However, *the benefit of BNNs for SNNs on reducing C_{mem} size and hence obtaining area, latency, power savings, without scarifying accuracy, has not been researched yet and this is the first work.*

Our novel contributions within this paper are:

- We reveal the impact of deploying error-resilient BNNs to analog-based SNN accelerator on the reduction of the membrane capacitor size C_{mem}.
- We demonstrate how the resiliency of BNNs, especially when they are trained in the presence of bit errors, leads to an increase in the robustness of SNN computation and enables significant C_{mem} reduction with marginal inference accuracy loss.
- We show that error resilient BSNNs achieve 50% reduction in C_{mem} size, two magnitudes of improvement in latency, and 57% in energy compared to the baseline (4-bit) SNN implementation, under minimal accuracy cost of 3% same as BNN.

II. Related Work

In [14], SNNs are acquired by training BNNs with the methods proposed by Hubara et al. [8]. After converting the BNNs to SNNs, simple design and run-time explorations are employed to achieve inference latency improvements. In [12], a method for training SNNs with binarized weights is proposed, without conversion from BNN to SNN.

The role of the membrane capacitor in analog-based implementations of SNNs has been identified in [7]. The study focuses on how the non-ideal characteristics of various memory technologies affect the requirement of the size of membrane capacitor. In [5], the authors evaluate SNNs on NOR flash computing array under input noise, relying on the inherent noise resiliency of NNs to sustain high accuracy. The capacitor size is set to a constant value for a the LeNET-5 NN model. The study in [15] surveys the resiliency properties of SNNs trained with various state-of-the-art algorithms and proposes an approach to train SNNs for resiliency. Their main focus is on different types synapse or neuron failures. The two studies in [16], [17] focus on fault analysis in hardware-implemented SNNs, assuming similar types of failures.

The above studies focus on the reliability issues that emerge from process variation, soft errors, and neuron and synapse faults. Methods for reducing membrane capacitor size have not been explored, although membrane capacitor size constitutes one of the major bottlenecks in analog SNNs, i.e. it determines energy, latency, area, and accuracy. To the best of our knowledge, we are the first to investigate that. In this work, we evaluate the impact of deploying error-resilient BNNs on analog implementation of SNNs, and formally show that for BNNs the membrane capacitor size and latency can be reduced significantly compared to multi-bit models. We then conduct extensive experiments to support this.

III. System Model

A. Binarized Neural Networks (BNNs)

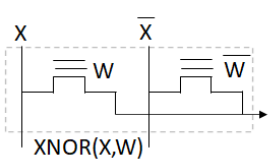
BNNs training: A common method for NN training applies stochastic gradient descent (SGD) with mini-batches. The training data is described with $\mathcal{D} = \{(x_1, y_1), \ldots, (x_I, y_I)\}$ with $x_j \in \mathcal{X}$ as the inputs, $y_j \in \mathcal{Y}$ as the labels, and $\mathcal{L}\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ as the loss function. $\mathbf{W} = (W^0, \ldots, W^L)$ are the weight tensors of layer $0, \ldots, L$ and $f_W(x)$ is the output of the NN. The goal is to find a solution for the optimization problem $\arg\min_W \frac{1}{I} \sum_{(x,y)\in\mathcal{D}} \mathcal{L}(f_W(x), y)$ by a mini-batch SGD strategy, computing gradients using backpropagation. To train BNNs, the weights and activations are binarized in the forward pass. For backpropagation, the floating point numbers are used for parameter updates [8].

BNNs inference: In BNNs, the weights and activations are binarized. Due to his, the MAC-operations of a layer can be computed as follows:

$$popcount(\text{XNOR}(\mathbf{W}_i^\ell, \mathbf{X}^{\ell-1})) > \mathbf{T}, \tag{1}$$

where *popcount* counts the number of '1's in the XNOR result, $\mathbf{W}_i^\ell$ describes the weights, $\mathbf{X}^{\ell-1}$ the inputs, ℓ the layer number, i the filter, and $\mathbf{T}$ is a learnable threshold parameter (attained by the batch normalization paramters [18]), whose comparison produces binarized values, which are fed to the subsequent hidden layer as inputs [8].

Error tolerance of BNNs: It has been shown that BNNs feature remarkable error resiliency under bit errors in the weights. Importantly, if errors are induced during training, the error resiliency can be increased even further [10]. To optimize BNNs for error resiliency, the state-of-the-art method for multiclass classification problems applies a modified hinge loss based on margin-maximization alongside error induction



X	W	$XW + \bar{X}\bar{W}$	XNOR(X,W)
0	0	0 + 1	1
0	1	0	0
1	0	0	0
1	1	1 + 0	1

Fig. 2: XNOR logic gate realization for binarized multiplication in SNNs.

during training [11]. The advantage of this method over the standard cross entropy loss has been evaluated for a variety of BNN models in [11].

B. Spiking Neural Networks (SNNs)

Stochastic input coding in SNNs: State-of-the-art SNN implementations use stochastic input coding for multiplication with multi-bit or binary weights. The stochastic input encoding is widely used [19], [20], [21], because it exploits the noise tolerance capability of NNs for computing efficiency. The input to the bit-line is the random variable following a Bernoulli distribution, with the firing rate $\frac{X_i}{X_{MAX}}$, where X_i is the input and X_{MAX} the maximum range of the input value in binary representation. The firing rate is interpreted as the probability for a spike. For the binary input case, the X_{MAX} is 1, and p_i is either 1 or 0.

Crossbar array for general multiplication: The typical operation of computing-in-memory is to compute logical AND between input and stored weights. The AND operation becomes binary multiplication when both input and weight are in binary representation. In our case, X_i is always binarized after stochastic conversion. This simplifies the multiplication to a certain number of AND operations determined by the number of binary cells to represent the weights.

XNOR-based crossbar for binarized SNNs: In this work, the XNOR array is adopted for implementation of BNNs. To realize XNOR in function, two memory cells (e.g. FeFET transistors, see [22]) are needed. Our XNOR design is based on the work in [23], see Fig. 2. The XNOR crossbar needs to be extended for the first layer, whose inputs are positive multiple bit. To realize the multiplication in this case, the method of subtracting NOT(W), inspired from [19] is adopted.

Analog implementation of SNNs: The implementation of SNNs using a memory crossbar and the input transformation is shown in Fig. 3. The design of memory crossbar with analog neuron circuit is based on [24]. The crossbar can implement both AND and XNOR operation, see Fig. 3. The steps of operation for SNNs are as follows:

(1) The input spikes are provided to the bit-lines of all memory cells in parallel. The binary operation results (XNOR or AND) in all word lines are computed in parallel as well. Finally, the output currents of all binary gates in a word line are accumulated by Kirchhoff's circuit law and passed to the neuron circuit.

(2) The accumulated output current charges the membrane capacitor C_{mem}. Note that, when stochastic input encoding

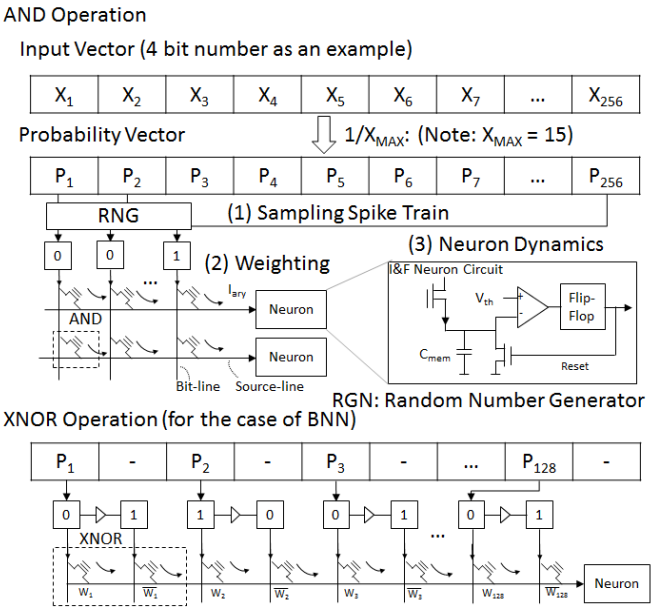


Fig. 3: The architecture design supporting both AND and XNOR operation.

is used, the same binary operation and accumulation in step 1 and 2 may need to be repeated for several cycles, with new input samples in each cycle, until the voltage across the capacitor surpasses the predetermined threshold voltage V_{th}. Once the voltage accross the membrane capacitor reaches V_{th}, the neuron generates an output spike. The observed frequency of the output spike is determined by the reciprocal of the time to first spike t_{fire} and is converted to an inner product by $\sum_i W_i X_i = X_{MAX} \frac{C_{mem} \cdot V_{th}}{I_{on}} \frac{1}{t_{fire}}$, where i denotes the index of different partial inner products and $I_{ON} = V_{BL} G_{cell}$ is the on-state current of the cell in the binary gate (G_{cell} is the conductance, V_{BL} the bitline voltage).

(3) The firing time for each neuron may be different. In the crossbar architecture, bit-line signals held for those neurons that are not fired yet still contribute current to fired neurons. To eliminate cell currents flowing to the fired neuron, the fired neuron will generate a signal to turn off cells on the word-line that are connected to the fired neuron.

(4) t_{fire} is acquired by a digital counter. Subsequently, the sampling number is converted to an inner product by a time-to-digit unit. Because of the limited crossbar size, the entire inner product computations need to be separated into a series of smaller inner products. For this reason, adders are needed for accumulation. After that, the result is ready for subsequent layer computations. As this follows conventional designs, the details of these devices are not shown in this work for brevity.

IV. Realizing BNNs on SNNs: Impacts of Binarization on SNNs

As shown in Fig. 4a, all binary inputs are sampled from the given probability at each cycle. The expected current, $E[I_{ary}]$, is equal to $I_{on} \sum_i W_i P_i$, where P_i is the normalized

input of X_i, W_i the binary weight, and I_{on} is ON-state cell current. Due to the stochasticity of the SNN computing scheme, sampled inner product currents can vary. To reduce discrepancy, multiple samples is required for averaging.

In analog computing SNNs, the sampled current contributes charges to membrane capacitor over time. The observed average current, $\widehat{E}[I_{ary}]$ is equal to total charges in the capacitor divided by the total charging time. When a neuron fires, it follows that the total charge in the capacitor has reached $C_{mem}V_{th}$. The observed average current can be obtained from

$$\widehat{E}[I_{ary}] = \frac{C_{mem}V_{th}}{t_{fire}}. \tag{2}$$

To reduce the discrepancy between observed average currents $\widehat{E}[I_{ary}]$, and system expected current $E[I_{ary}]$, a certain number of samples is required based on the law of large numbers. The way to increase the number of samples is by increasing C_{mem} size such that more samples of inner product currents are required to make the neuron fire.

In short, the inner product precision is a function of the membrane capacitor size, which needs to be configured based on the number of required samples to achieve a certain inference accuracy. In the following, we will discuss the implications of binarization and resiliency, focusing on membrane capacitor size reduction and its effects on analog SNN accelerators, with respect to energy and latency.

A. Impact of Binarization in SNNs on Membrane Capacitor

When deploying BNNs as SNNs, the inputs of the BNN are bi-state, which is either 1 or 0. The probability of binary input is also either 1 or 0. In other words, the inputs are the *deterministic* values, and the output current is a constant value. Taking the advantage of the deterministic output when deploying BNNs as SNNs, the SNN does not need multiple cycles to collect multiple output current for achieving high precision output, unlike multi-bit SNNs. That is, the stochastic behavior of SNN is eliminated for any size of membrane capacitor selected. Therefore, in the case of binarized weights and inputs, it is possible to reduce the size of the membrane capacitor. In the following, we focus on two impacts of binarization on SNN computation. First, we discuss the one-cycle computation stemming from binarization and why this allows reduction of C_{mem} size. Then, we explain why the accumulated charges are expected to be smaller with binarization.

Binarization allows Cmem reduction: The computing error in stochastic sampling needs to be small enough to prevent inference accuracy loss [19], and we describe it as $\mathrm{VAR}(\widehat{E}[I_{ary}^{stoch}]) \leq K$, where $K > 0$ is a certain model-dependent criterion, which needs to be defined at design or run time to sustain a certain SNN inference accuracy. To reduce $\mathrm{VAR}(\widehat{E}[I_{ary}^{stoch}])$, the sampling number S needs to be increased, since the variance becomes smaller with a higher number of samples S due to the law of large numbers, i.e. $\mathrm{VAR}(\widehat{E}[I_{ary}^{stoch}]) \propto \frac{1}{S}$. The sample number depends on $S = \frac{C_{mem}^{stoch}V_{th}}{I_{on}} \cdot \frac{1}{\sum_i W_i P_i} \cdot \frac{1}{\tau}$, and since the on-state current I_{on} and the targeted inner product value $\sum_i w_i p_i$ can be replaced with $\widehat{E}[I_{ary}^{stoch}] = I_{on}\sum_i W_i P_i$, we can write

$$\widehat{S} = \frac{C_{mem}^{stoch}V_{th}}{\widehat{E}[I_{ary}^{stoch}]} \cdot \frac{1}{\tau} \geq Q,$$

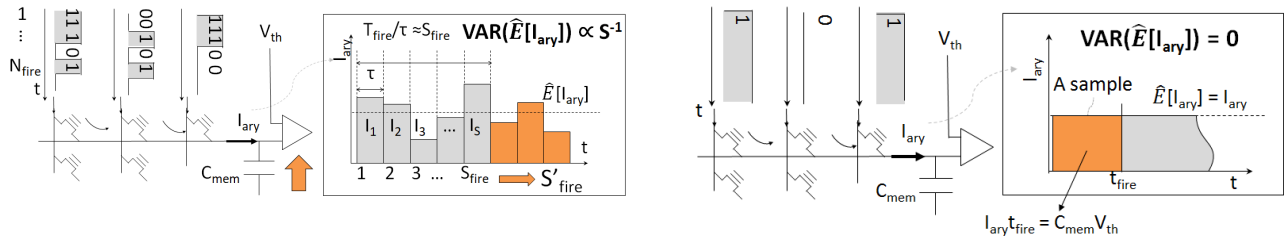
where $\tau > 0$ is the pulse width, which limited by circuit parasitics. For sustaining accuracy, the number of samples S needs to high enough, i.e. $S > Q$ such that the variance has an upper bound $\mathrm{VAR}(\widehat{E}[I_{ary}^{stoch}]) < K$ for any inner inner product value. The number of minimum samples Q must be found empirically based on the NN model for the stochastic input case. For BNNs, there is no bound Q, since $S = 1$. For BNNs, S can be set to 1 because $\mathrm{VAR}(\widehat{E}[I_{ary}^{bin}]) = 0$, since the input is deterministic. Due to this, in BNNs, no Q needs to be found, and the variance is independent from the size of the membrane capacitor. *Thus, in BNNs, the membrane capacitor size can be reduced without accuracy cost stemming from a low number of samples.*

When there is a minimal requirement for precision of the sampled current, in the stochastic case, more charges (proportional to sample number S) may be needed compared to the binarized case. We assume there is a constraint, the minimum charge unit $q^{min} = I_{ON}\tau$, for storing accumulated currents in the membrane capacitor, where I_{ON} is the cell current and τ is the pulse width determined by circuit parasitics. In the stochastic case, $\sum_s \sum_i q^{min}$ will be stored in the capacitor. In the binarized case, this is $\sum_i q^{min}$, since only one sample is neccassary. *Charge needs to be accumulated equal or more times in the stochastic case than in the binarized case*, i.e. $\sum_i q^{min} \leq \sum_s \sum_i q^{min}$. This implies that a larger membrane capacitor size is needed in the stochastic case, to hold the charge. When the capacitor size is set to be smaller in BNNs, the voltage across the capacitor increases faster, causing earlier spikes. For the stochastic case, the required charge is larger, and the required spiking time is also increased. However, the firing times vary based on the distribution of the inner product. There are still minimum and maximum firing times, which are model-dependent. Although firing time is related to latency, it cannot be used as a metric for measuring and comparing latency between binarized and stochastic case, since the slowest firing time of neurons determines latency.

B. Impact of Binarization in SNNs on Latency

As described above, the latency of a neuron depends on the inner product value, determining t_{fire}. However, for different number of samples S and number of bits used for encoding weights or inputs, the distribution and the range of the inner product may differ greatly. This is why it is difficult to compare t_{fire} among different models and sizes of C_{mem}. To fairly compare latency among different cases, we propose to a metric we call guaranteed response time (GRT).

The guaranteed response time (GRT) $log(t_G)$ is an upper bound for the latency of a neuron, as illustrated in Fig. 5. We define the GRT as the maximum time that is given to a neuron, within which it must respond with a spike, i.e. the maximum allowed latency of a neuron is set to t_G. The neurons with



(a) I_{ary} over time for stochastic input NNs.

(b) I_{ary} over time for BNNs.

Fig. 4: (a) I_{ary} over time for stochastic input NNs. For each sample of inputs, the I_{ary} may differ. For precise current estimation, a certain number of samples is required. As a result, for the stochastic input case, VAR($\widehat{E}[I_{ary}]$) is inversely proportional to the sampling number. (b) I_{ary} over time for BNNs. The inputs are deterministic. Thus, the I_{ary} does not differ among samples. I_{ary} will always be equal to $\widehat{E}[I_{ary}]$, since VAR($\widehat{E}[I_{ary}]$)=0.

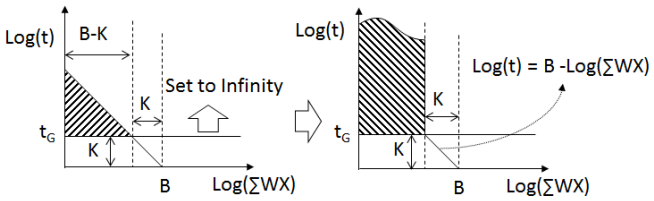


Fig. 5: Description of information loss metric.

$log(t_{fire}) > t_G$ are considered as no-fire and their t_{fire}s are set to infinity (inner product 0). Once the neurons are considered as no-fire, the information of those neuron is lost. The information loss metric is the available region ratio of output value defined by $\frac{B-K}{B}$. The relation between t_{fire} and $\sum_i W_i X_i$ is defined as: $t_{fire} = \frac{C_{mem} V_{th}}{I_{ON}} \cdot \frac{X_{MAX}}{\sum_i W_i X_i}$. For convenience, we take the logarithm, which results in: $log(t_{fire}) = log(\frac{C_{mem} V_{th} X_{MAX}}{I_{ON}}) - log(\sum_i W_i X_i)$. For values with $\log(\sum W_i X_i) >$ B-K of inner product values is lost. Thus, we define the information loss metric as:

$$\frac{B-K}{B} = 1 - \frac{K}{B} = 1 - \frac{t_G}{log(\frac{C_{mem} X_{MAX} V_{th}}{I_{ON}})} \quad (3)$$

where B and K are labels in Fig. 5, $\frac{1}{X_{MAX}}$ is unit of input x, and 10^{t_G} is provided gaurantee response time.

Our metric describes the information loss for an arbitrary guaranteed response time of 10^{t_G}. It is evident in Eq. (3) that the metric is indexed by the product of $C_{mem} X_{MAX}$, which implies that a *smaller membrane capacitor leads to higher information loss*. To minimize information loss, C_{mem} should have a large value. However, when a low value of the information metric is desired, the guaranteed response time needs to be set to a large value, affecting the latency.

C. Impact of Binarization in SNNs on Energy

The required energy to make the neuron fire is reduced with smaller membrane capacitor size. The energy consumption of analog-based SNNs is mainly determined by the synapse array, neuron circuit, and analog-to-digital conversion. When emerging devices with steep sub-threshold slope are employed, the static energy consumption of an amplifier in a neuron circuit is eliminated by a single transistor [25]. Furthermore, the conversion from t_{fire} to a digital representation only needs to be performed at firing time. This means, the energy consumption is mainly determined by the current that is produced by memory arrays performing inner products.

The consumed energy of the crossbar array is proportional to the charge coming out from the memory array as shown in Eq. (4). Charge from the memory array is stored in the memory capacitor and increases the membrane voltage. The neuron only fires when the required charge reaches $C_{mem} V_{th}$. Therefore, with a reduction of the membrane capacitor size, the energy consumed from array is also reduced.

$$E = V_{BL} V_{th} C_{mem} \quad (4)$$

As shown in (4), due to $E = V_{BL} \int_0^{t_{fire}} I_{ary} dt$ and $\int_0^{t_{fire}} I_{ary} dt = V_{th} C_{mem}$, the energy of the systems depends on bitline voltage V_{BL}, threshold voltage V_{th}, and membrane capacitor size C_{mem}. V_{BL} is set such that reduction of V_{BL} would cause further reliability issues. The required capacity for charges corresponds to the factor of $C_{mem} V_{th}$. The scalability of the charge capacity depends on the size of C_{mem}, while V_{th} has limited operation margin in an analog circuit. Thus, in this work, the V_{th} is set to a constant, and the requirement of the charge capacity is indexed by the size of C_{mem}.

D. Errors by Cmem Reduction and Countermeasures

Errors from information loss: With a small capacitor size, parts of the inner product may not be considered during computation, because of the same charge, the voltage increases faster, causing earlier spikes. This error can be described with $t_{fire}^{IL} = t_{fire} - \epsilon_{IL}$ due to earlier firing time. Using a small GRT has the same effect, since it reduces t_{fire} by a constant value. For controlling the information loss, the capacitor size needs to be set accordingly.

Errors from limited sampling frequency: With smaller firing times caused by smaller C_{mem} size, the sensing frequency of the FF needs to be increased. Consider the case of one rising edge at time t_{re} and one spiking signal at t_{fire}. The spiking signal is not latched until the arrival of the subsequent rising edge. The sensing error is then described by $t_{re} - t_{fire}$ (see Fig. 6). The error manifests itself as a neuron timing

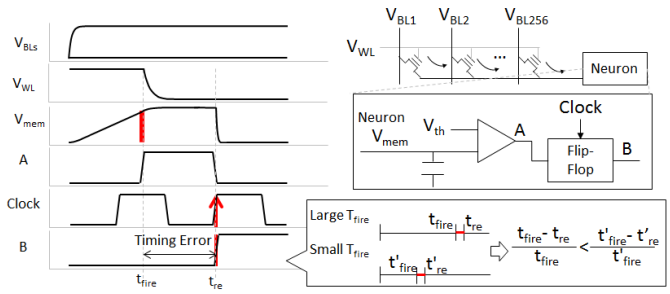


Fig. 6: Waveform of SNN circuit. Timing error by FF, since it cannot latch the spike. The smaller t_{fire}, the larger the ratio of timing error $\frac{t_{re}-t_{fire}}{t_{fire}}$. The max. timing error is bounded by the sensing period, while t_{fire} is increased over time. Thus, the increase of t_{fire} will reduce the timing error ratio.

error, i.e. the shifted firing time is $t^*_{fire} = t_{fire} + \epsilon_{SF}$. To alleviate this, the sampling frequency of the FF needs to be increased accordingly with smaller capacitor size, to minimize ϵ_{SF}. However, f_{SF} cannot be set arbitrarily high due to fundamental limits.

Combined effect of errors and countermeasures: We denote the error due to reduced capacitor size, with errors from information loss (IL) and from limited sampling frequency (SF) as

$$t^{IL,SF}_{fire} = t_{fire} - \epsilon_{IL} + \epsilon_{SF}. \quad (5)$$

These errors may degrade inference accuracy significantly. To tolerate these errors, we use the method for resiliency optimization in [11], where bits of the weights W_i are flipped during training. Due to the property of XNOR, each flipped weight bit induces an output error (see Fig. 2). These bit flip induced errors change the popcount-result by ϵ_{FLIP}. For this weight-flipping scheme, we can describe the errors with $\sum_i W_i X_i \pm \epsilon_{FLIP} > T$. Due to the similarity of the weight errors to IL and SF errors, we expect that the method for increasing general error resiliency of BNNs in [11], i.e. using the modified hinge loss in combination with bit flip injection with a certain flipping probability p_{FLIP}, allows reduction of capacitor size in SNNs with minimal accuracy cost.

V. Evaluation

To evaluate the impact of binarization in SNNs, we consider as a baseline a stochastic input NN with multi-bit weights, given C_{mem} size, and a certain inference accuracy. Our goal is to evaluate to what extent binarization combined with error resiliency optimization can reduce capacitor size, energy, and latency, while sustaining inference accuracy.

A. Experiment Setup

For evaluation, we use the FashionMNIST dataset to prove our primary concept. The model used in this work is a CNN with three layers (C64-MP2-C64-MP2-FC2048-FC10), see [11]. In the case of BNNs, the A-functions returns binarized values, while in the 4-bits weights baseline model, ReLU is used. Both the BNNs and 4-bits NNs have the same architecture, except for the activation function and weight precisions.

Circuit Parameters per Crossbar (256x256) @ 65nm Technology Node		
Component	Spec	Energy (pJ)
Counter [28]	1.4 pJ/cycle	1.4 GRT/25ns
Adder (8 bits) [29]	0.16 pJ/spike	40
Crossbar+Neuron	$2V_{BL}V_{th}C_{mem}$ pJ/spike	$512V_{BL}V_{th}C_{mem}$
Time-to-digit [30]	107.5 pJ/spike	27520
RNG (8bits) [31]	135.76 pJ/cycle	135.76 GRT/25ns

TABLE I: Energy configurations of SNN macro.

SNN Model	SF=1GHz	SF=2GHz
4-bits NN	15.8pF	15.8pF
BNN	12.6pF	10pF
ER-BNN	10pF	7.9pF

TABLE II: C_{mem} size to sustain inference accuracy of ≥ 0.88

For the BNNs, we use the modified hinge loss (MHL) with the hyperparameter $b = 128$ (see [11]). We use BNNs that were trained in two separate ways. One BNN was trained only with the MHL.The other BNN was optimized for error resiliency by injecting a probability of error $p_{FLIP} = 10\%$ per binary weight. We use batch sizes of 256 and initial learning rates of 10^{-3}. The learning rate is decreased exponentially every 25 epochs by 50 percent. We used the Adam optimizer [26] for 200 training epochs. For the 4-bit model, which serves as the baseline to compare the binarized SNNs to, we use quantization-aware training. The number of training epochs is 33, as it reaches full convergence earlier. The initial learning rate is 0.01, which we decay by 0.1 every tenth epoch. We use the the Adam optimizer here as well.

The stochastic model of SNNs for the binarized and 4-bit case is built from samples of circuit simulation. Samples are pairs of fire time and inner product generated from randomly-generated input and weight vectors. The stochastic model is provided to the SNN evaluation tool. The crossbar size is set to be 256×256 which can operate 128 inputs in parallel.

The memory technology used for evaluation is FeFET [27]. The cell architecture is simulated based on the measured data. The design scenario of high polarization (PR) and coercive field (EC) is adapted because of its high ON-OFF ratio. In this design scenario, the ON-state cell current is 10uA, operating at V_g = 0.6V with the ON-OFF ratio over 1000.

The circuit energy configurations are listed in Table I. The role of each component is described in Subsec. III-B.

B. Experiment Results

In the following, we first reduce the membrane capacitor in BNNs and 4-bits NNs, without employing any countermeasures, and show that the errors (from information loss and small sensing frequency) become large and drastically impact inference accuracy. As the first countermeasure, we increase the sensing frequency to reduce the sensing errors. We then evaluate to which extent the membrane capacitor can be reduced when we fully exploit the error resiliency of ER-BNNs. Finally, we compare latency and energy consumption

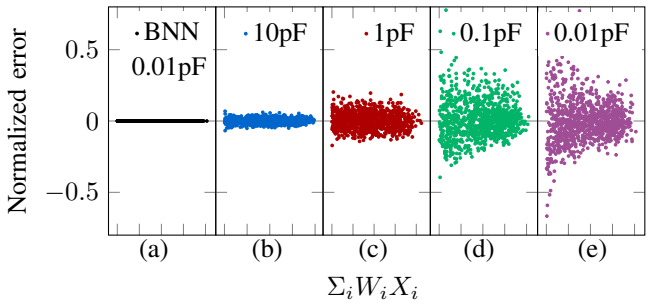
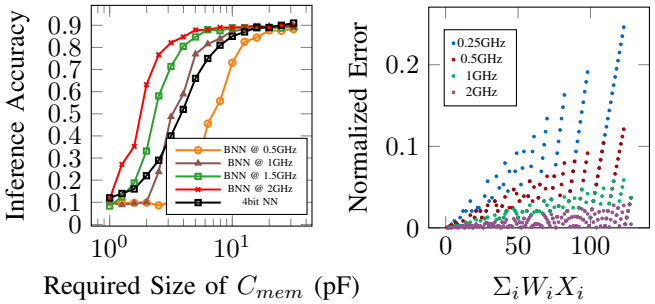


Fig. 7: Normalized error from information loss over inner product for different membrane capacitor sizes. (a) shows errors for BNN, while (b) - (e) shows errors for 4-bits model. The sampling frequency is set to infinity to avoid interference.



(a) (b)

Fig. 8: Effect of sensing frequency (SF). (a) C_{mem} for BNN can be reduced by increasing SF. It is not possible for 4-bits model (information loss) (b) The SFs are varied for analyzing effects of sensing errors in BNN. Increasing SF reduces the normalized error $\frac{t_{re}-t_{fire}}{t_{fire}}$. Capacitor size is 10 pF.

of the 4-bits model, BNN and ER-BNN with maximum capacitor size reduction under an accuracy goal.

Information loss: To ideally have no error interference by limited sampling frequency, in Fig. 7, we set the sampling frequency to infinity. In the figure, we show the errors from information loss (see Sec. IV-D) for the BNNs in subplot (a) and for the 4-bit case in subplots (b) - (e). We observe that in BNNs, the errors from information loss are always zero in the tested cases. This is due to the small charge in the capacitor when BNNs are run. We do not show the errors for all capacitor sizes for BNNs, since the errors are always zero. In the 4-bits case ((b) - (e)), the errors from information loss get larger when the capacitor size is reduced. There are larger charges in the 4-bits case than in BNNs, since in the 4-bits case, multiple samples are needed to estimate the inner product. Combined with a small membrane capacitor, early firings will be induced, leading to information loss.

Capacitor optimization under sensing frequency: Here, we focus on errors caused by limited sampling frequency for the case of BNNs, since BNNs do not experience errors from information loss from small capacitor and therefore provide a better initial point for optimization. In Fig. 8a we show how increasing the sensing frequency in BNNs from 1 GHz to 2 GHz allows capacitor size reduction while sustaining accuracy at a high level. In Fig. 8b, we show that the normalized computing error in BNNs can be reduced by increasing sensing frequency. The reason is the increase of sensing frequency, which reduces the upper bound of timing error, as described in Fig. 6.

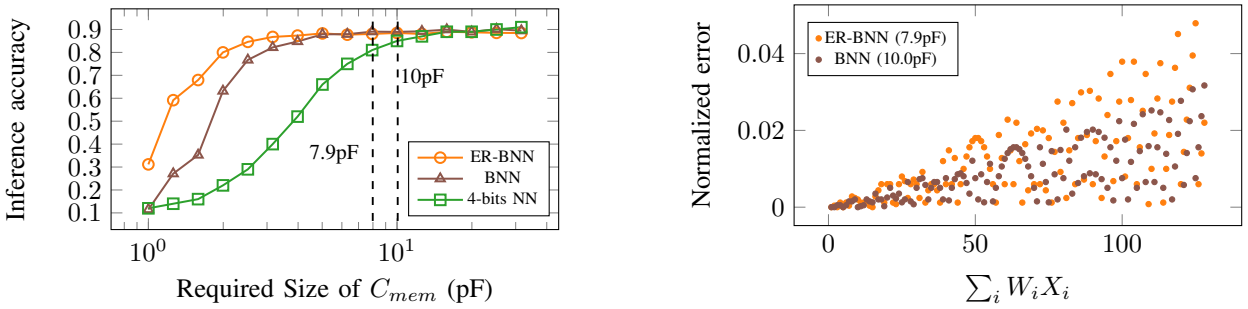
Capacitor optimization under ER-BNNs: Although deploying BNNs on SNNs can reduce the size of membrane capacitor with high sensing frequency, there are fundamental limits, which we consider here to be 2 GHz. To enable further reduction of membrane capacitor size without increasing sensing frequency, we aim to exploit the error resiliency of BNNs to tolerate the sensing errors. We show the benefit of ER-BNNs in Fig. 9a. In the ER-BNN, the capacitor size can be reduced by 20% compared to the BNN model without ER. In Fig. 9b, we plot the sensing frequency error for the case in which a capacitor of size 7.9 pF is selected such that the accuracy is above 0.88. Although the sensing errors in the case with 7.9 pF are larger than with 10pF, the ER-BNN can tolerate the them. As a result, the classification accuracy is sustained for smaller capacitor sizes, where the size of C_{mem} for ER-BNN is 50% smaller than that for the 4-bits model.

Latency improvement: The guaranteed response time (GRT) over accuracy is shown in Fig. 10a. The BNN can sustain accuracy with smaller GRT than the 4-bits model, while the ER-BNN can sustain high accuracy with smaller GRT than the BNN with no ER. In Fig. 10b, the GRT for the 4-bits model is higher. The reason is that a lower GRT causes more information loss than BNNs. In some cases, the expected firing time may exceed GRT, which is assumed to be a no-fire. The 4-bits model cannot sustain accuracy in these cases due to high information loss. As a summary, the GRT of the different models are shown in Fig. 11a. The BNN has two orders of magnitude in GRT improvement compared to the 4-bits model, while the ER-BNN shows 20% GRT improvement compared to the BNN without ER.

Energy saving: The energy configuration of our considered system is shown in Tab. I. The energy used in each model is as shown in Fig. 11b. The result indicates that the required energy to generate a spike in BNNs without ER is 36.7% less than 4-bits model, while the ER-BNN can reduce by 57% of energy compared to the 4-bits model.

VI. Conclusion

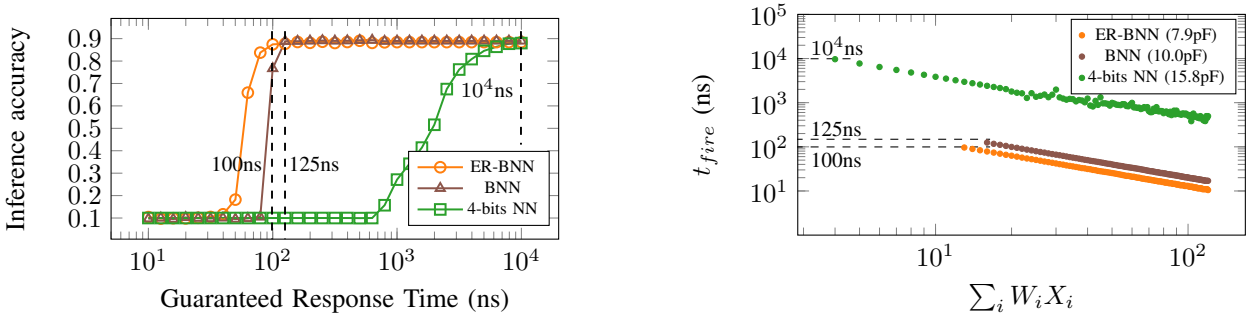
We studied the impact of deploying error-resilient BNNs (ER-BNNs) on analog implementations of SNNs. We focused on the reduction of the membrane capacitor size, since it constitutes one of the major bottlenecks in analog SNNs, determining inference accuracy, energy usage, latency, and area. Through analyzing the properties of analog SNN circuits, we showed that binarization allows capacitor size reduction with less accuracy cost than in multi-bit SNNs, since binarization leads to deterministic inputs and less capacitor charge. We also established the connection between the latency and



(a) Requirement of C_{mem} size for training schemes

(b) Normalized error

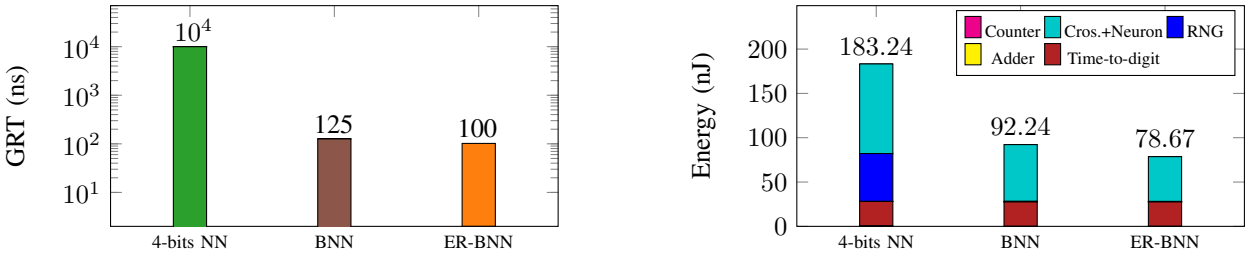
Fig. 9: Noise tolerance with sensing frequency (SF) 2GHz. (a) The ER-BNN can tolerate smaller C_{mem} than other models. (b) With C_{mem} selected from (a), satisfying accuracy ≥ 0.88, the ER-BNN can tolerate more normalized timing error, $\frac{t_{re}-t_{fire}}{t_{fire}}$, induced from smaller C_{mem}.



(a) Effects of guarantee response time

(b) t_{fire} plots considering guarantee response time

Fig. 10: Comparisons of guaranteed response time (GRT). (a) The minimal GRT of BNN is two orders of magnitude smaller than 4-bits model. (b) The GRT is selected from (a) satisfying accuracy ≥ 0.88. The ER-BNN can sustain accuracy despite errors from reduced GRT and limited sensing frequency (SF). All SF are set to 2GHz.



(a) Guaranteed Response Time (GRT) for Models

(b) Energy for 256 Neurons in a Macro Generating their Own Spike

Fig. 11: Summary of computing efficiency for each models. The minimal size of membrane capacitor for each case is selected to sustain the classification accuracy. The ER-BNN can hold the classification of ≥ 0.88 with the smallest GRT.

the membrane capacitor size, from which we deduced that binarization allows higher latency reduction than in multi-bit cases. We also showed that the energy consumption depends on the membrane capacitor size. However, the membrane capacitor size reduction comes at the cost of timing errors, for which we developed a model for evaluating the trade-off between membrane capacitor size and the SNN inference accuracy. As a countermeasure to the errors, we optimize BNNs for ER to tolerate the errors while sustaining high accuracy. In our experiments, we evaluated the trade-off between the capacitor size reduction and the errors. Our results indicate that, compared to 4-bits SNNs, deploying ER-BNNs as SNNs leads to 50% and 57% reduction of capacitor size and energy respectively, and two orders of magnitude in improvement in latency, while high inference accuracy is sustained.

ACKNOWLEDGMENTS

This paper has been supported by Deutsche Forschungsgemeinschaft (DFG) project OneMemory (Nr.: 405422836), by the Collaborative Research Center SFB 876 "Providing Information by Resource-Constrained Analysis" (project number 124020371), subproject A1 (http://sfb876.tu-dortmund.de), and also supported in part by research grants from the Ministry of Science and Technology of Taiwan (MOST-107-2923-E-001-001-MY3, MOST-109-2221-E-002-147-MY3), National Taiwan University (NTU-110L880603).

REFERENCES

[1] M. Bouvier, A. Valentian, T. Mesquida, F. Rummens, M. Reyboz, E. Vianello, and E. Beigne, "Spiking neural networks hardware implementations and challenges: A survey," *J. Emerg. Technol. Comput. Syst.*, vol. 15, Apr. 2019.

[2] H. Mostafa, "Supervised learning based on temporal coding in spiking neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 7, pp. 3227–3235, 2018.

[3] I.-M. Comşa, K. Potempa, L. Versari, T. Fischbacher, A. Gesmundo, and J. Alakuijala, "Temporal coding in spiking neural networks with alpha synaptic function: Learning with backpropagation," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2021.

[4] T. Liu, Z. Liu, F. Lin, Y. Jin, G. Quan, and W. Wen, "Mt-spike: A multilayer time-based spiking neuromorphic architecture with temporal error backpropagation," in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 450–457, IEEE, 2017.

[5] Y. Xiang, P. Huang, R. Han, C. Li, K. Wang, X. Liu, and J. Kang, "Efficient and robust spike-driven deep convolutional neural networks based on nor flash computing array," *IEEE Transactions on Electron Devices*, vol. 67, no. 6, pp. 2329–2335, 2020.

[6] S. Dutta, C. Schafer, J. Gomez, K. Ni, S. Joshi, and S. Datta, "Supervised learning in all fefet-based spiking neural network: Opportunities and challenges," *Frontiers in Neuroscience*, vol. 14, p. 634, 2020.

[7] M.-L. Wei, H. Amrouch, C.-L. Sung, H.-T. Lue, C.-L. Yang, K.-C. Wang, and C.-Y. Lu, "Robust brain-inspired computing: On the reliability of spiking neural network using emerging non-volatile synapses," in *2021 IEEE International Reliability Physics Symposium (IRPS)*, pp. 1–8, 2021.

[8] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Advances in neural information processing systems*, pp. 4107–4115, 2016.

[9] A. Laborieux, M. Ernoult, T. Hirtzlin, and D. Querlioz, "Synaptic metaplasticity in binarized neural networks," *Nature Communications*, vol. 12, no. 2549, 2021.

[10] T. Hirtzlin, M. Bocquet, J. Klein, E. Nowak, E. Vianello, J. M. Portal, and D. Querlioz, "Outstanding bit error tolerance of resistive ram-based binarized neural networks," in *International Conference on Artificial Intelligence Circuits and Systems, AICAS*, pp. 288–292, 2019.

[11] S. Buschjäger, J.-J. Chen, K.-H. Chen, M. Günzel, C. Hakert, K. Morik, R. Novkin, L. Pfahler, and M. Yayla, "Margin-maximization in binarized neural networks for optimizing bit error tolerance," *DATE '21*.

[12] S. Lu and A. Sengupta, "Exploring the connection between binary and spiking neural networks," *Frontiers in Neuroscience*, vol. 14, Jun 2020.

[13] S. R. Kheradpisheh, M. Mirsadeghi, and T. Masquelier, "Bs4nn: Binarized spiking neural networks with temporal coding and learning," 2020.

[14] D. Kim, G. Kim, C. S. Hwang, and D. S. Jeong, "ewb: Event-based weight binarization algorithm for spiking neural networks," *IEEE Access*, vol. 9, pp. 38097–38106, 2021.

[15] C. D. Schuman, J. Parker Mitchell, J. T. Johnston, M. Parsa, B. Kay, P. Date, and R. M. Patton, "Resilience and robustness of spiking

[21] M. Koo, G. Srinivasan, Y. Shim, and K. Roy, "Sbsnn: Stochastic-bits enabled binary spiking neural network with on-chip learning for energy efficient neuromorphic computing at the edge," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 8, pp. 2546–2555, 2020.

neural networks for neuromorphic systems," in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10, 2020.

[16] E.-I. Vatajelu, G. Di Natale, and L. Anghel, "Special session: Reliability of hardware-implemented spiking neural networks (snn)," in *2019 IEEE 37th VLSI Test Symposium (VTS)*, pp. 1–8, 2019.

[17] T. Spyrou, S. A. El-Sayed, E. Afacan, L. A. Camuñas-Mesa, B. Linares-Barranco, and H.-G. Stratigopoulos, "Neuron Fault Tolerance in Spiking Neural Networks," in *DATE 2021*, (Grenoble (virtuel), France), Feb. 2021.

[18] E. Sari, M. Belbahri, and V. P. Nia, "How does batch normalization help binary training?," *arXiv:1909.09139*, 2019.

[19] T. Hirtzlin, B. Penkovsky, M. Bocquet, J.-O. Klein, J.-M. Portal, and D. Querlioz, "Stochastic computing for hardware implementation of binarized neural networks," *IEEE Access*, vol. 7, pp. 76394–76403, 2019.

[20] A. Sengupta, G. Srinivasan, D. Roy, and K. Roy, "Stochastic inference and learning enabled by magnetic tunnel junctions," in *2018 IEEE International Electron Devices Meeting (IEDM)*, pp. 15–6, IEEE, 2018.

[22] K. Ni *et al.*, "Ferroelectric ternary content-addressable memory for one-shot learning," *Nature Electronics*, vol. 2, no. 11, pp. 521–529, 2019.

[23] X. Chen, X. Yin, M. Niemier, and X. S. Hu, "Design and optimization of fefet-based crossbars for binary convolution neural networks," in *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1205–1210, 2018.

[24] T. Tang, L. Xia, B. Li, R. Luo, Y. Chen, Y. Wang, and H. Yang, "Spiking neural network with rram: Can we use it for real-world application?," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 860–865, IEEE, 2015.

[25] C. Lee, J. Sung, and C. Shin, "Understanding of feedback field-effect transistor and its applications," *Applied Sciences*, vol. 10, no. 9, p. 3070, 2020.

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

[27] K. Ni, A. Gupta, O. Prakash, S. Thomann, X. S. Hu, and H. Amrouch, "Impact of extrinsic variation sources on the device-to-device variation in ferroelectric fet," in *2020 IEEE International Reliability Physics Symposium (IRPS)*, pp. 1–5, IEEE, 2020.

[28] Y.-W. Kim, J.-S. Kim, J.-H. Oh, Y.-S. Park, J.-W. Kim, K.-I. Park, B.-S. Kong, and Y.-H. Jun, "Low-power cmos synchronous counter with clock gating embedded into carry propagation," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 56, no. 8, pp. 649–653, 2009.

[29] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.

[30] V. Guidotti, G. Paim, L. M. Rocha, E. Costa, S. Almeida, and S. Bampi, "Power-efficient approximate newton–raphson integer divider applied to nlms adaptive filter for high-quality interference cancelling," *Circuits, Systems, and Signal Processing*, vol. 39, pp. 5729–5757, 2020.

[31] B. Perach *et al.*, "An asynchronous and low-power true random number generator using stt-mtj," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 11, pp. 2473–2484, 2019.