Weight Mapping Properties of a Dual Tree Single Clock Adiabatic Capacitive Neuron

Mike Smart, Sachin Maheshwari, *Member, IEEE*, Himadri Singh Raghav, *Member, IEEE*, Alexander Serb, *Senior Member, IEEE*

Abstract—Dual Tree Single Clock (DTSC) Adiabatic Capacitive Neuron (ACN) circuits offer the potential for highly energyefficient Artificial Neural Network (ANN) computation in full custom analog IC designs. The efficient mapping of Artificial Neuron (AN) abstract weights, extracted from the software-trained ANNs, onto physical ACN capacitance values has, however, yet to be fully researched. In this paper, we explore the unexpected hidden complexities, challenges and properties of the mapping, as well as, the ramifications for IC designers in terms accuracy, design and implementation. We propose an optimal, AN to ACN methodology, that promotes smaller chip sizes and improved overall classification accuracy, necessary for successful practical deployment. Using TensorFlow and Larq software frameworks, we train three different ANN networks and map their weights into the energy-efficient DTSC ACN capacitance value domain to demonstrate 100% functional equivalency. Finally, we delve into the impact of weight quantization on ACN performance using novel metrics related to practical IC considerations, such as IC floor space and comparator decision-making efficacy.

Index Terms—Adiabatic Logic, Binary Neural Networks, Switched Capacitor, Analog Computation, Energy-Efficiency.

I. INTRODUCTION

DIABATIC LOGIC (AL) is a low-power ASIC design technique where charge used for computation is periodically returned to a slowly alternating AC power supply [1], [2]. This process allows for energy-efficient analog computation, compared to non-adiabatic, CMOS digital solutions that use fixed DC supplies. AL requires a compatible supply source for charge recovery, such as those designed using an inductor-based oscillator [3]–[5] or a capacitive-based step-charging circuit [6]–[8]. AL has been applied to the design of in-memory computation [9], content addressable memories [10], power drivers [11], secure hardware design [12], error detection in data transmission [13] and spiking networks [14], [15].

In parallel to advances in AL, ANNs have been implemented in analog hardware using Switched Capacitor (SC) networks [16]–[20]. Early work in non-adiabatic SC neural networks, such as by Tsividis *et al* [21], used a proportional mapping scheme to convert AN weights to SC capacitance values. Other authors have used non-adiabatic charge-to-weight-to-capacitance value relationships [16], switch capacitance to

This work has been, in part, funded by Defence Science and Technology Laboratory (Dstl), UK.

M. Smart, S. Maheshwari, H. S. Raghav, A. Serb, are with the Centre for Electronics Frontiers, School of Engineering, University of Edinburgh, Scotland, EH9 3FB, UK. (E-mail: {msmart2, maheshwari.sachin, hraghav, aserb}@ed.ac.uk.

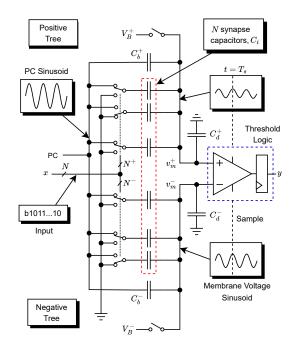


Fig. 1. Double Tree Single Clock (DTSC) N-bit differential ACN [25] with binary input vector, \mathbf{x} , two capacitive trees with a total of N switched synapse capacitors and a single sinusoidal Power Clock (PC) supply. A Threshold Logic (TL) unit generates a binary output, y, by sampling, and comparing, the two membrane voltages, v_m^{\pm} , at a time, T_s .

amplifier feedback capacitance ratios [22], binary weighted mappings [17], [23] and, most recently, differential capacitance values [15]. The Adiabatic Capacitive Artificial Neuron (ACAN) combined AL techniques with an SC circuit to minimize use of energy dissipating resistive elements and perform low-power, AN, dot-product computation [5], [24]. ACAN was restricted to ANNs with positive-valued weights and a binary activation function. Similar to ACAN, the adiabatic DTSC ACN was subsequently introduced with support for both positive and negative weights using only a minimal number of switched capacitors [25].

Away from the developments in SC and AL analog hardware, there has also been significant recent research into Quantized Neural Networks (QNNs) [26] and Binary Neural Networks (BNNs) that use ANs with a binary activation function [27], [28]. Generally, work in this area has been driven by the desire for fast, efficient and low power digital AN hardware. Historically, training BNNs was problematic due to the requirement to back-propagate error gradients through neurons with binary outputs. The introduction of the

Straight Through Estimator (STE) and STE variants [26], [29], however, has meant that training BNNs to perform real-world tasks, such as image classification, is now possible with open-source BNN software frameworks, such as TensorFlow-based Larq [30]. Researchers have also shown the ability of BNNs, with suitable resources, to achieve classification rates comparable to standard ReLU-based ANNs on well-known datasets, such as MNIST and CIFAR-10 [26]. This has opened the door for the mapping of real-world, software-trained ANs directly onto AL-based SC hardware, like the DTSC ACN.

At first glance, mapping weights from these real-world ANNs onto ACN capacitance values, using a simple proportional scheme, as before, might seem trivial. In this paper we show that, actually, a more elaborate approach is required. The paper introduces an optimal mapping that provides exact functional equivalence, whilst using a minimal number of switched capacitors. Further, we show that ANN training, quantization and mapping can have significant impact on the ultimate hardware performance, such as IC classification accuracy and chip size. The paper also introduces a range of mapping-related tools and properties to aid the IC designer.

The paper begins in Section II with a brief ACN operational review. Section III then considers various approaches to mapping AN weights and details the mathematical properties and significant benefits of the preferred conditional mapping algorithm. Section IV discusses the practical ASIC design implications of the conditional mapping, which have not been previously explored in the literature. In Section V the paper concludes with an ACN mapping demonstration using various MNIST-trained ANNs, BNNs and ONNs.

II. ACN OPERATION

To understand how an abstract artificial neuron can efficiently map onto adiabatic hardware first requires an examination of the operation of an ACN. The DTSC ACN circuit [25], reproduced in Fig. 1, uses a sinusoidal voltage supply, referred to as a Power Clock (PC), to charge two independent banks, or trees, of synapse capacitors on the upswing of the PC. Two sinusoidal membrane voltages, v_m^{\pm} , one for each tree, appear on the input terminals of a Threshold Logic (TL) unit. The TL then samples and compares both membrane voltages at a time, $t = T_s$, typically when the sinusoidal PC voltage, $V_{pc}(t)$, reaches its peak voltage, V_{max} . The binary output of the TL, y, is ideally set to binary '1' if $v_m^+ \geq v_m^-$, else it is set to binary '0'. On the PC downswing, the charge used for computation is returned to the supply. The DTSC ACN charge recovery principle remains the same as with previous ACAN work. A more detailed description of the cyclical operation can be found in [5], [25].

The magnitudes of v_m^{\pm} are determined by which synapse capacitors are allowed to contribute in each tree. This is controlled by a set of N Single Pole Double Throw (SPDT) switches, shown to the left of the synapse capacitors in Fig. 1. Each SPDT switch is controlled by a single bit from an N-bit binary input signal, x. If the ith input bit, x_i , is binary '1' then the PC is allowed to propagate to the associated synapse capacitor C_i and contribute to the membrane voltage.

Otherwise the synapse capacitor is connected to ground. More formally, the N synapse capacitors are split into two disjoint sets of N^+ capacitance values, C_i^+ where $i \in I^+$, and N^- capacitance values, C_i^- where $i \in I^-$. I^\pm are disjoint subsets of the indexing set $I = \{0.., N-1\}$, such that $N = N^+ + N^-$.

In addition to the synapse capacitors, there are two extra capacitors on each tree: a bias capacitor, C_b , and a ballast capacitor, C_d . The bias capacitors always allow the PC to propagate, providing a minimal swing of the membrane voltage, even when all inputs are zero. The ballast capacitors can be used to control the maximum swing of v_m^{\pm} .

Finally, each capacitive tree has a DC bias voltage, V_B . Before computation the ACN is put into an initial reset state by closing the two Transmission Gate (TG) switches connected to V_B^{\pm} . This initializes both membrane voltages to a known fixed value. Once the reset is complete, these TG switches are re-opened to allow for computation.

Based on the previous definitions, and using standard capacitive voltage division, it is possible to formulate an idealized equation for v_m^{\pm} , at a time t, on each tree as follows

$$v_m^{\pm}(t) = V_B^{\pm} + V_{pc}(t) \sum_{i \in I^{\pm}} \frac{C_i^{\pm} x_i + C_b^{\pm}}{C_T^{\pm} + C_b^{\pm} + C_d^{\pm}}$$
 (1)

where C_T^{\pm} represents the summed, parallel, synapse capacitance values in each tree. The membrane voltages in (1) can be alternatively expressed more simply as

$$v_m^{\pm}(t) = V_B^{\pm} + V_{pc}(t) \frac{C_{on}^{\pm}}{C_{on}^{\pm} + C_{off}^{\pm}}$$
 (2)

where C_{on}^{\pm} is the sum of the switched on capacitance values $(x_i = 1)$ plus the bias capacitance in each tree, and C_{off}^{\pm} is the sum of all the switched off capacitance values $(x_i = 0)$ connected to ground, plus the ballast capacitance, C_{d}^{\pm} .

Several differences exist between the previous ACAN design and the DTSC ACN. There are now two capacitor trees along with bias capacitors. Also, there is no TL absolute DC reference voltage and the Single Pole Single Throw (SPST) synapse switches have been replaced by SPDT switches. Importantly, the number of synapse capacitors remains as N in DTSC compared to other SC signed-weight solutions using 2N differential capacitance values [15].

III. AN TO ACN MAPPING

In order to replicate the mathematical operation of a software-trained AN, the weights and biases need to be mapped to the capacitance values in the DTSC ACN circuit. The mapping requires that the same binary output, y, from the software AN and the hardware ACN, is generated for the same binary input vector, \mathbf{x} , for all possible valid input sequences.

For AL charge recovery to work, a high impedance, binarizing comparator is used in the ACN TL. This architecture is inherently well-suited for mapping ANs that utilize binary activation functions. A compatible AN accepts \mathbf{x} as input and generates a single bit output, y, as such

$$y = \begin{cases} 1, & \text{if } \mathbf{w} \cdot \mathbf{x} = \sum_{i \in I} w_i x_i \ge \tau \\ 0, & \text{otherwise} \end{cases}$$
 (3)

where w_i are N synapse weights that form a weight vector \mathbf{w} and τ is a bias value. The weights and bias may be real-valued $(w_i, \tau \in \mathbb{R})$, quantized or binary in nature.

The N weights can be split into two sets of N^+ positive-valued weights, w_i^+ , and N^- negative-valued weights, w_i^- . To perform the mapping, the positive-valued w_i^+ , need to be converted to a set of N^+ positive-valued ACN capacitance values, C_i^+ in the positive tree of the DTSC ACN. Similarly, the negative-valued w_i^- , need to be mapped to N^- positive-valued capacitance values, C_i^- , in the negative tree.

A. Conditional mapping

To map software AN functionality to an ACN, the dot-product condition defined in (3) must be preserved. As such, a $conditional\ mapping$ approach is proposed that solely focuses on whether the ACN will output a binary '1' when the relative condition $v_m^+ \geq v_m^-$ is met, rather than absolute voltage levels. Assuming $V_B^+ = V_B^-$ then (1) can be expressed as

$$V_{pc}(t) \left[\sum_{i \in I^{+}} \frac{C_{i}^{+} x_{i} + C_{b}^{+}}{C_{A}^{+}} \right] \ge V_{pc}(t) \left[\sum_{i \in I^{-}} \frac{C_{i}^{-} x_{i} + C_{b}^{-}}{C_{A}^{-}} \right]$$
(4)

where $C_A^\pm = C_T^\pm + C_b^\pm + C_d^\pm$. The ACN will output binary '0' when this condition is not met. The benefit of SPDT switches in ACN means that both C_A^\pm terms remain constant, regardless of input. This, significantly, provides an input-output linearity, not present in ACAN, such that (4) can be refactored by multiplying both sides by C_A^-/C_T^- , collecting bias terms, multiplying the LHS by C_T^+/C_T^+ and canceling $V_{pc}(t)$

$$\frac{C_T^+ C_A^-}{C_T^- C_A^+} \sum_{i \in I^+} \frac{C_i^+ x_i}{C_T^+} \ge \frac{C_b^-}{C_T^-} - \frac{C_b^+ C_A^-}{C_T^- C_A^+} + \sum_{i \in I^-} \frac{C_i^- x_i}{C_T^-}$$
 (5)

Returning to the AN model, condition (3) can be split into two positive and negative components and rewritten as

$$\sum_{i \in I^{+}} w_{i}^{+} x_{i} \ge \tau + \sum_{i \in I^{-}} |w_{i}^{-}| x_{i} \tag{6}$$

Then by replacing the now positive-valued weighting terms on both sides of condition (6) with

$$|w_i^{\pm}| = C_i^{\pm} w_T^{\pm} / C_T^{\pm} \tag{7}$$

where w_T^{\pm} is a positive-valued weight vector scaling factor

$$w_T^{\pm} = \sum_{i \in I} |w_i^{\pm}| \tag{8}$$

a software-based AN mapping condition is derived, which can be mapped directly onto the hardware condition (5).

$$\frac{w_T^+}{w_T^-} \sum_{i \in I^+} \frac{C_i^+ x_i}{C_T^+} \ge \frac{\tau}{w_T^-} + \sum_{i \in I^-} \frac{C_i^- x_i}{C_T^-} \tag{9}$$

Comparing the AN condition (9) with ACN condition (5):

$$\frac{C_T^+ C_A^-}{C_T^- C_A^+} = \frac{w_T^+}{w_T^-} \tag{10}$$

$$\frac{C_b^-}{C_w^-} - \frac{C_b^+ C_A^-}{C_w^- C_+^+} = \frac{\tau}{w_w^-} \tag{11}$$

Under the condition $C_T^+/C_T^-=w_T^+/w_T^-=C_T/w_T$, where $w_T=w_T^++w_T^-$ and $C_T=C_T^++C_T^-$, then (10) simplifies to the balanced capacitive tree condition $C_A^-=C_A^+$. The synapse capacitance values can then be determined from (7) as

$$C_i^{\pm} = C_T |w_i^{\pm}| / w_T \tag{12}$$

and (10) can be expanded and rewritten in terms of C_d^- as

$$C_d^- = \frac{(w_T^+ - w_T^-)C_T}{w_T} + C_b^+ + C_d^+ - C_b^- \tag{13}$$

which holds for $C_d^- \geq 0$ when $w_T^+ \geq w_T^-$ and

$$C_d^- = \frac{(w_T^+ - w_T^-)C_T}{w_T} + C_b^+ \text{ and } C_d^+ = C_b^-$$
 (14)

Rewriting (10) in terms of C_d^+ , with the same conditions, then $C_d^+ \geq 0$ holds when $w_T^- \geq w_T^+$ and

$$C_d^+ = \frac{(w_T^- - w_T^+)C_T}{w_T} + C_b^- \text{ and } C_d^- = C_b^+$$
 (15)

From (11) the condition $C_b^- \ge 0$ holds for $\tau \ge 0$ when

$$C_b^+ = 0 \text{ and } C_b^- = \tau C_T / w_T$$
 (16)

as τ , C_T and w_T^{\pm} are all positive-valued by definition. Similarly, by using the C_A ratio in (10), the condition $C_b^+ \geq 0$ holds for $\tau < 0$ when

$$C_b^- = 0 \text{ and } C_b^+ = |\tau| C_T / w_T$$
 (17)

We now have the foundations necessary to calculate all the physical ACN capacitance values based on AN model parameters. Under ideal conditions the ACN will produce an identical output as the AN, given the same binary input vector, \mathbf{x} . The value of C_T can be considered as a design choice, and a *scaling constant* of the physical ACN.

The proposed conditional mapping scheme has demonstrated that the process is non-trivial, requiring specific values of C_d^{\pm} for a robust mapping. The mapping also raises a number of important points that we will explore, in detail, in the remainder of this paper: 1) the conditional mapping ensures AN and ACN output equivalency but does not define the relationship between the ACN v_m^\pm voltages and the originating AN $\mathbf{w} \cdot \mathbf{x}$ dot product; 2) there is no indication whether the mapping is in some sense optimal e.g. in terms of minimal C_d ; 3) from a mapping perspective, the bias capacitors could be treated as always-on capacitors; 4) a new design choice, C_T , has been introduced but with no guidance on what value a designer should use; and lastly, 5) although C_A^{\pm} in (4) is constant due to the choice of SPDT synapse switches and provides linearity with respect to the input, x, it remains a function of C_i and, as such, places a normalizing constraint on the computation not present in the software AN.

B. Vector space

To better understand the points discussed in III-A this section examines the AN to DTSC ACN conditional mapping in \mathbb{R}^N vector weight space. To aid visualization, a simple N=2 vector space with $\tau=0$ is considered, as demonstrated in Fig. 2. In this case, the AN output, y, as specified by

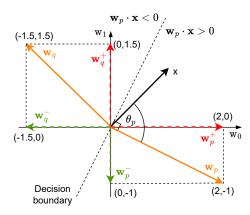


Fig. 2. N=2 vector space with input vector, $\mathbf{x}=(1,1)$ and two example weight vectors ($\mathbf{w}_{\mathbf{p}}$ and $\mathbf{w}_{\mathbf{q}}$) that occupy \mathbb{R}^2 . Each weight vector (orange) represents a different AN, each of which can be resolved into two vectors: a vector with positive-valued components (red) and a vector with all negative-valued components (green).

(3), is '1' when $\mathbf{w} \cdot \mathbf{x} = |\mathbf{w}| |\mathbf{x}| \cos \theta \ge 0$. This occurs when the angle, θ , between \mathbf{w} and \mathbf{x} satisfies $-90^{\circ} \le \theta \le 90^{\circ}$. So, in this AN implementation the weight vector magnitude, $|\mathbf{w}|$, is irrelevant to the output, y, and only θ needs to be considered. Consequently, only the direction of \mathbf{w} needs to be preserved in the AN to ACN mapping to generate an equivalent y.

To match the dual tree structure of the target DTSC ACN architecture requires that the weight vector, \mathbf{w} , be resolved into two orthogonal \mathbb{R}^N vectors, \mathbf{w}^+ and \mathbf{w}^- , representing all the positive-valued and all the negative-valued weights respectively, such that $\mathbf{w} = \mathbf{w}^+ + \mathbf{w}^-$ and $\mathbf{w}^+ \cdot \mathbf{w}^- = 0$. More formally, $\mathbf{w}^+ = (w_0^+, ... w_i^+ ... w_{N-1}^+)$ where $w_i^+ = w_i$ if $w_i > 0$ else 0, and conversely, $w_i < 0$ for \mathbf{w}^- .

Each of these \mathbf{w}^{\pm} vectors now needs to be normalized by w_T^{\pm} . This is required to match the normalizing constraint generated by ACN capacitive voltage division in each DTSC tree. Next, as the negative-valued weights are required to map to positive-valued physical capacitance values, \mathbf{w}^- is flipped to generate a new vector, $\mathbf{w}^- = -1\mathbf{w}^-$. Together, normalization and flipping, constrains the two resulting vectors to an N-1 dimensional subspace defined by a $\mathbb{R}^N_{\geq 0}$ hyperplane, as demonstrated in Fig. 3. This hyperplane intersects all N unit vectors in the weight space. This means both of these modified vectors are now in the positive-only quadrant, such that $\mathbf{w}^+/w_T^+ \in \mathbb{R}^N_{\geq 0}$ and $\mathbf{w}^-/w_T^- \in \mathbb{R}^N_{\geq 0}$. Orthonormality of the modified vectors seen in Fig. 3 is unlikely for N>2 due to the often non-sparse nature of trained weight vectors.

Although now in a suitable form for DTSC mapping, the resulting cumulative vector $\mathbf{w}^+/w_T^+ - \mathbf{\overline{w}}^-/w_T^-$ does not point in the same direction as \mathbf{w} , due to the differing normalization scalars, w_T^\pm , typical in trained ANs. To correct the direction a positive-valued scalar value, δ , can be added to one of the normalizing terms. This scaling preserves a weight form that can still map to the two DTSC voltage division trees. In the Fig. 3 example, as $w_T^+ > w_T^-$, the negative term is appropriate as δ can only scale down the vector magnitude. This correction term means, with a specific value of δ , that $\mathbf{w}' = \mathbf{w}^+/w_T^+ - \overline{\mathbf{w}}^-/(w_T^- + \delta)$ will point in the same direction as \mathbf{w} . From the proportional scaling properties of the two similar triangles

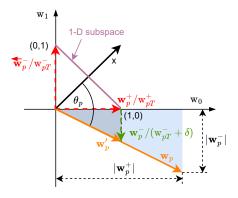


Fig. 3. Example weight vector $\mathbf{w_p}$ mapped onto a DTSC compatible weight space where the normalized weight vectors are constrained to a positive-valued N-1 dimensional subspace (purple). A reconstructed vector, $\mathbf{w_p'}$, can be made to point in same direction (θ_p) as the original $\mathbf{w_p}$ through vector subtraction and use of a correction term, δ . Note that $|\mathbf{w_p'}| \neq |\mathbf{w_p}|$.

shown in blue in Fig. 3, it is known that the ratio of the two triangles lengths are equivalent. Specifically, $|w^+|/w_T^+|w^+|=|w^-|/(w_T^-+\delta)|w^-|$ and, as such, $\delta=w_T^+-w_T^-$. As will be shown later, this also holds when N>2.

Returning to the ACN, the N capacitance values, C_i , can hypothetically be expressed as a capacitive vector, \mathbf{c} , with signed values. Using proportional weight scaling then $\mathbf{c}=\alpha\mathbf{w}$, where α is a positive-valued scalar and, as such, $\mathbf{c}^\pm=\alpha\mathbf{w}^\pm$ with $C_T^\pm=\alpha w_T^\pm$. As $\tau=0$ we let $C_b^\pm=0$ and assuming $C_d^+=0$ the ACN differential membrane voltage $\Delta v_m=v_m^+-v_m^-$ when $V_{pc}(t)=V_{max}$ can be expressed as

$$\Delta v_m = V_{max} \left[\sum_{i \in I^+} \frac{C_i^+ x_i}{C_T^+} - \sum_{i \in I^-} \frac{C_i^- x_i}{C_T^- + C_d^-} \right]$$
(18)

or, in vector notation with the input resolved as $\mathbf{x} = \mathbf{x}^+ + \mathbf{x}^-$

$$\Delta v_m = V_{max} \left[\mathbf{c}^+ \cdot \mathbf{x}^+ / C_T^+ - \overleftarrow{\mathbf{c}}^- \cdot \mathbf{x}^- / (C_T^- + C_d^-) \right] \quad (19)$$

or by substitution and given that $\mathbf{c}^+ \cdot \mathbf{x}^- = \overleftarrow{\mathbf{c}}^- \cdot \mathbf{x}^+ = 0$

$$\Delta v_m = V_{max} \left[\mathbf{c}^+ \cdot \mathbf{x} / C_T^+ - \overleftarrow{\mathbf{c}}^- \cdot \mathbf{x} / (C_T^- + C_d^-) \right]$$
 (20)

This means Δv_m is proportional to $\mathbf{C} \cdot \mathbf{x}$, where \mathbf{C} is a hypothetical normalized capacitance value vector defined by

$$\mathbf{C} = \mathbf{c}^+ / C_T^+ - \overleftarrow{\mathbf{c}}^- / (C_T^- + C_d^-) \tag{21}$$

Like \mathbf{w}' the vector \mathbf{C} will preserve the direction of \mathbf{c} , and thus \mathbf{w} , when the correction $C_d^- = C_T^+ - C_T^-$ or $C_T(w_T^+ - w_T^-)/w_T$, is used, as previously shown algebraically. Repeating the process when $w_T^- > w_T^+$ is a trivial exercise.

C. Capacitive dot-product

As shown, the AN weight vector, \mathbf{w} , can be conditionally mapped to the DTSC ACN as positive-valued capacitance values. However, although the direction of \mathbf{w} and \mathbf{C} are equivalent, their magnitudes tend to differ and generally $|\mathbf{w}| \neq |\mathbf{C}|$ and, as such, $\mathbf{w} \cdot \mathbf{x} \neq \mathbf{C} \cdot \mathbf{x}$. Continuing with $\tau = 0$, the sampled TL differential input voltage is given by $V_{max}\mathbf{C} \cdot \mathbf{x}$ or

$$\Delta v_m = V_{max} |\mathbf{C}| |\mathbf{x}| \cos \theta \tag{22}$$

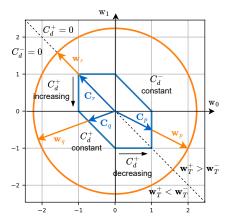


Fig. 4. Three examples of N=2 weight vectors $(\mathbf{w_p}, \mathbf{w_q}, \mathbf{w_r})$ all with the same magnitude, $|\mathbf{w}| = \sqrt{5}$, but differing directions, ϕ . Also shown are the corresponding conditionally-mapped capacitance vectors $(\mathbf{C_p}, \mathbf{C_q}, \mathbf{C_r})$. The ballast capacitor sizes, C_{d}^{\pm} , are also shown to vary with ϕ .

Thus, both the magnitude, $|\mathbf{C}|$, and relative direction, θ , of the capacitive vector, \mathbf{C} , may have practical implications.

An N=2 weight vector with a constant magnitude, $|\mathbf{w}|$, and varying absolute direction, ϕ , will trace out a circular path, as shown in Fig. 4. Due to normalization the path of \mathbf{C} is, however, not circular. In this superficial \mathbb{R}^2 example the hexagonal path, shown in blue, for \mathbf{C} is traced. In these non-typical cases, where $N^{\pm}=1$, one tree will include a single capacitor with no ballast, making the weight value ineffective. Typically, the path when $N^{\pm}>2$ will form a hypercube, with $|\mathbf{C}|$ being a function of ϕ . This implies the capacitive dot-product, and consequently, Δv_m , will change, perhaps unexpectedly, for different ACNs, dependent on ANN training. It should be noted with conditional mapping, that the same \mathbf{C} path will be traced, regardless of $|\mathbf{w}|$.

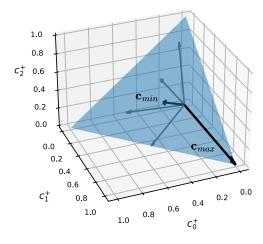


Fig. 5. $N^+=3$ space with six example vectors constrained to triangular 2D plane (light blue). Vector \mathbf{c}_{min} has minimum length when $c_0^+=c_1^+=c_2^+$ and vector \mathbf{c}_{max} is longest when all weights are zero bar one.

Fig. 5 shows a $\mathbb{R}^3_{>0}$ vector space with six example \mathbf{c}^+/C_T^+ vectors constrained to a 2D surface. Weight vectors from trained ANs will vary in their distributions. An AN could include small numbers of large-valued weights combined

with numerous smaller-valued weights. These distributions can have $|\mathbf{c}^+/C_T^+|$ closer to the maximum of 1. Conversely, $|\mathbf{c}^+/C_T^+|$ will be minimum when the vector is orthogonal to the hyperplane, or when $C_i^+ = C_T^+/N^+ \, \forall i \in I^+$ which gives a magnitude of $1/\sqrt{N^+}$. Thus, Δv_m will become increasingly small with increasing N^+ , and potentially, further reduced in the case of binarized weights.

Finally, from the Δv_m definition in (18) it is clear that $-1 \leq \mathbf{C} \cdot \mathbf{x} \leq 1$. This condition can potentially lead to saturation of $\mathbf{C} \cdot \mathbf{x}$ compared to the sinusoidal $\mathbf{w} \cdot \mathbf{x}$ as ϕ varies. However, this is increasingly rare as N^{\pm} increases and, as it is only the sign of the dot product that impacts the binary output, can generally be ignored.

D. Ballast capacitance size

As seen in Fig. 4, and more clearly in Fig. 6, the size of C_d^\pm will vary with the original weight direction, ϕ . The conditional mapping equations (14) and (15) show that as $w_T^\pm \to 0$ then $C_d^\pm \to C_T$. In the unlikely extreme, where all weights are all positive or all negative, then C_d^\pm will be zero or C_T .

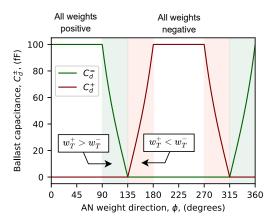


Fig. 6. Variation in C_d^\pm where N=2 and $\tau=0$ with varying ϕ and $C_T=100fF$. Ballast size is independent of the weight magnitude, $|{\bf w}|$.

Fig. 7 shows the more general case for an N-bit ACN $(\tau=0)$ where only an \mathbb{R}^2 subspace spanned by the orthogonal \mathbf{c}^\pm vectors is shown. The value of C_d^- can be determined, as before, using the scaling properties of similar triangles, with $|c^-|/(C_T^- + C_d^-)|c^-| = |c^+|/C_T^+|c^+|$ which gives $C_d^- = C_T^+ - C_T^-$. Fig. 7 shows that the conditional mapping generates the largest possible $|\mathbf{C}|$ for a given \mathbf{w} . Smaller $|\mathbf{C}|$ will occur when there is ballast capacitance on both trees.

To summarize, the conditional mapping approach from the perspective of Δv_m and C_d can be considered optimal, but these values remain dependent on the original weight vector direction. It is worth a look at some alternative mappings.

E. ReLU and alternate DTSC mappings

An alternative mapping is possible by considering that (10) also holds for $w_T^+ \geq w_T^-$ when

$$C_d^+ = w_T^- C_T / w_T + C_b^- \text{ and } C_d^- = w_T^+ C_T / w_T$$
 (23)

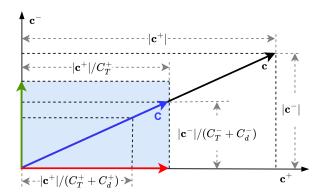


Fig. 7. The \mathbb{R}^2 subspace spanned by the orthogonal \mathbf{c}^\pm vectors in \mathbb{R}^N . The hypothetical normalized capacitive vector will be constrained to the area shaded in blue. Different values of C_d^\pm are capable of preserving the direction of the scaled capacitive vector, $\mathbf{c} = \alpha \mathbf{w}$. The $\tau = 0$, conditionally-mapped, \mathbf{C} is shown with $\max |C|$ and $\min \sum C_d^\pm$.

which also works for the case where $w_T^+ < w_T^-$. Using both ballast capacitors is perhaps a more **balanced mapping**, but as indicated in Fig. 7, the total capacitance, $\sum C_d^{\pm}$, is much higher $(\geq C_T)$ than the conditional mapping inferring more energy and chip size, as well as, smaller-valued Δv_m .

For IC designers considering physical layout or energy, it may be appropriate to differentiate between the switched C_i and the non-switched capacitors. However, mathematically it is simpler to incorporate the bias, τ , into an N+1 weight vector with $w_N=-\tau$ and $x_N=1$. When $w_{N+1,T}^+\geq w_{N+1,T}^-$ a conditional vectored-bias mapping is defined with

$$C_d^- = \frac{(w_{N+1,T}^+ - w_{N+1,T}^-)C_{N+1,T}}{w_{N+1,T}} \text{ and } C_d^+ = 0$$
 (24)

One of the ballast capacitors is now a no-fit, which removes a capacitor and overheads. Yet, as will be seen later, there are practical reasons for instantiating both C_d^{\pm} and to use a non-vectored C_b mapping. Regardless, both approaches to mapping τ , the bias can be included in \mathbf{C} and (22) will hold.

As seen previously, $\mathbf{C} \cdot \mathbf{x} \neq \mathbf{w} \cdot \mathbf{x}$ with conditional mapping. This is generally fine when using a binary activation function. However, for completeness and comparison, it is possible to match the AN and ACN dot products, as would be required for a ReLU activation function. Using a vectored-bias approach with $dim(\mathbf{x}) = dim(\mathbf{w}) = dim(\mathbf{C}) = N+1$ a **ReLU mapping** that satisfies $\mathbf{C} \cdot \mathbf{x} = \mathbf{w} \cdot \mathbf{x}$ can be achieved as follows

$$V_{max} \left[\frac{\mathbf{c}^+ \cdot \mathbf{x}^+}{C_T^+ + C_d^+} - \frac{\mathbf{c}^- \cdot \mathbf{x}^-}{C_T^- + C_d^-} \right] = \mathbf{w} \cdot \mathbf{x} = \mathbf{w}^+ \cdot \mathbf{x}^+ - \overleftarrow{\mathbf{w}}^- \cdot \mathbf{x}^-$$
(25)

Considering each term separately

$$V_{max} \frac{\mathbf{c}^{\pm} \cdot \mathbf{x}^{\pm}}{C_T^{\pm} + C_d^{\pm}} = \mathbf{w}^{\pm} \cdot \mathbf{x}^{\pm}$$
 (26)

With linear vector scaling $\mathbf{c}^{\pm} = C_T^{\pm} \mathbf{w}^{\pm} / w_T^{\pm}$

$$\frac{V_{max}C_T^{\pm}}{w_T^{\pm}} \frac{\mathbf{w}^{\pm} \cdot \mathbf{x}^{\pm}}{C_T^{\pm} + C_d^{\pm}} = \mathbf{w}^{\pm} \cdot \mathbf{x}^{\pm}$$
 (27)

$$\frac{V_{max}C_T^{\pm}}{w_T^{\pm}} = C_T^{\pm} + C_d^{\pm} \tag{28}$$

$$C_d^{\pm} = \frac{C_T}{w_T} (V_{max} - w_T^{\pm}) \tag{29}$$

For the ballast capacitors to be realizable in the ReLU mapping the conditions $w_T^{\pm} \leq V_{max}$ must hold. This places a restriction on the trained weight vector. Also, in this mapping there is a dependency on the absolute voltage, V_{max} .

IV. MAPPING PROPERTIES AND IC CONSIDERATIONS

This section discusses the various mathematical properties of the mapped DTSC ACN circuit. We also discuss some of the practical considerations to an IC designer considering incorporating adiabatic DTSC ACNs into an ASIC design.

A. Computational accuracy

Mapping ACNs requires migrating numerical representations of weights in software to physical IC capacitance values. This infers all the standard device manufacturing inaccuracies, such as mismatch [31]–[33], process and thermal variations [34], [35], technology constraints, noise sources, and unwanted parasitic capacitive effects, especially on the v_m node that do not occur in software ANs.

Standard technology library, Metal-Insulator-Metal (MIM) and custom Metal-Oxide-Metal (MOM) capacitor designs in full custom ICs allow implementation of both real-valued and quantized capacitance values. Technology constraints, such as the granular geometry restrictions imposed by the chip manufacturer, can possibly introduce small quantization errors. However, careful quantization during AN weight training can mitigate for these inaccuracies to an extent.

Parasitic capacitance on the v_m nodes, as shown in Fig. 8, especially when instantiating small-valued capacitors for the process node used, have the potential to significantly impact computation. The accumulated parasitic capacitances to ground, $C_{A_{par}}^{\pm}$, from all ACN capacitors connected to v_m , can be non-negligible. A solution is to use tool-generated estimates for $C_{A_{par}}^{\pm}$ and use parasitic-compensated ballast capacitance values, as $C_{A_{par}}^{\pm}$ is in parallel with C_d^{\pm} . An issue may occur if $C_{A_{nar}}^{\pm} > C_d^{\pm}$ or $C_d = 0$, which is addressed later.

The efficacy of the TL comparator also has a direct impact on output accuracy, especially when resolving smallvalued Δv_m . Designing comparators with low-offset across temperature ranges and process corners is possible [25] but factors, such as task complexity, N, mapping or TL input voltage range, can all potentially compact the computation into a smaller Δv_m region around the decision boundary, pushing the comparator towards offset-generated errors. In this paper, an instability metric, Ψ , is defined as the fraction of all input samples processed that generate a $|\Delta v_m|$ less than some defined voltage tolerance level. The metric Ψ can aid an IC designer in regard to required comparator accuracy and possible additional error rates. Use of a non-zero software bias, τ , when training is generally useful for ACN-compatible ANNs when inputs such as x = 0 are a valid application use case. As we shall see weight quantization during training can also be a useful tool. Integrating offset compensation into the mapping is a possibility, but offset is a function of many

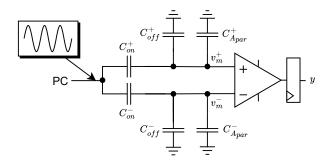


Fig. 8. Simplified DTSC representation for a specific input. Parallel synapse and bias capacitors connected to the PC in each tree are replaced with a single capacitor, C_{on}^{\pm} , and parallel synapse and ballast capacitors connected to ground in each tree are replaced with a single capacitor, C_{off}^{\pm} , as defined by (2). Total parasitic capacitances on each v_m node, C_{nna}^{\pm} , is introduced.

parameters including temperature, layout symmetry and input capacitive loading, which could vary from ACN to ACN.

Detailed error analysis of TL comparator low-offset design, impacts of capacitor quantization, SPDT switch parasitics including RC delays and the C_d parasitic-compensation process are beyond the scope of this paper.

B. Voltage swing range

The input PC signal, $V_{pc}(t)$, will ideally swing between 0V and a peak value of V_{max} . As such, the two membrane voltages will also swing from 0V to some modulated peak value. Assuming no delays and an ideal comparator sampling at $t=T_s$ then v_m will swing in the range

$$\frac{V_{max}C_b^{\pm}}{C_T^{\pm} + C_b^{\pm} + C_d^{\pm}} \le v_m^{\pm}(t = T_s) \le \frac{V_{max}(C_T^{\pm} + C_b^{\pm})}{C_T^{\pm} + C_b^{\pm} + C_d^{\pm}} \quad (30)$$

The swing in each tree will be minimal when $x_i=0 \, \forall i$ and maximized when $x_i=1 \, \forall i$ and dependent on C_b^\pm and C_d^\pm . Ballast capacitor scaling has a dominant effect on the upper limit. When $C_d=0$, then the maximal swing will be V_{max} . The bias capacitors have a significant effect on the lower limit.

C. PC voltage scaling

The single PC clock design in DTSC means computation, as defined in (4), is independent of $V_{pc}(t)$. This is a hugely beneficial differential property of DTSC that means $V_{pc}(t)$ magnitude can theoretically be reduced in design, or dynamically post-design, without affecting ACN output. Lower $V_{pc}(t)$ implying decreased energy consumption [25].

Real PC generator circuits are far from ideal and drops in $V_{pc}(t)$ magnitude over time and inaccurate sampling are very possible, where $V_{pc}(t=T_s) < V_{max}$. These non-intentional drops in PC, and consequently Δv_m , should have no effect on the result. This robustness is a significant advantage of DTSC over previous architectures like ACAN, where the use of absolute reference voltages offers no protection to these realities. In practice, other considerations and circuit design constraints on TL, increased inaccuracies with reduced Δv_m , switch and supply behavior will limit PC scaling.

D. Symmetry

There is no natural symmetry in trained, real-world ANNs with $N^+ \neq N^-$ and $w_T^+ \neq w_T^-$ being common. However, in the conditionally-mapped DTSC ACN, some useful symmetry exists regardless. In the mapping derivation in section III-A it was shown $C_A^+ = C_A^-$, which states that the total capacitance in each tree is equal. This symmetry is relevant for designers when performing layout of ACN capacitive arrays in custom ICs, ensuring a natural capacitive balance. Furthermore, any unwanted parasitic capacitances that are proportional to each C_A^\pm will also be, in principle, easier to balance.

E. Capacitive pillars

The symmetry property of DTSC ACN trees means that fixed and equal amounts of capacitance added to the bias, or ballast, in each tree will computationally cancel out. We refer to these fixed values C_{pb} and C_{pd} as bias and ballast capacitive *pillars*, respectively. Capacitive pillars are a design tool that allow an IC designer to control the minimum and maximum swing of the membrane voltages without theoretically affecting computation. This is useful for ensuring the computation occurs in the voltage range best suited to the type of comparator used, which may be narrower than that of the PC.

Capacitive pillars also resolve the situation where mapped bias and ballast capacitance values fall below the minimum allowable capacitance value, C_{min} , for the technology in use. Capacitance values below C_{min} will not be physically realizable. For example, given mapped capacitance values of $C_b^-=30fF$ and $C_b^+=0fF$ for a MIM capacitor 180nm technology where $C_{min}=35fF$ it is possible to add a 35fF capacitive pillar to the bias capacitors i.e. $C_b^-=65fF$ and $C_b^+=35fF$ which are now both valid capacitance values for the technology and computation remains unaffected. The same process allows for parasitic-compensated ballast capacitors, ensuring that the compensated ballast is positive-valued.

The downside of pillars is that they reduce the dynamic range of Δv_m . Further, bias pillars can be used to generate an absolute voltage offset for the comparator. However, a need to generate an absolute voltage will break the voltage scaling property defined in IV-C, this use should be avoided if possible, favoring comparators that are capable of working at lower differential voltages.

F. Capacitive scaling and C_T design choice

Weight quantization in QNNs and BNNs has reduced resources resulting in faster, lower footprint and power in digital processors [26], [27]. A primary driver for the ACN is energy-efficient AN computation. Here AN weights are mapped to physical capacitors, which require physical IC space affecting overall chip area, device cost and energy.

The ACN capacitive voltage divider trees are scalable. Multiplying all capacitor values in (4) by a constant value has no affect on computation, as the constant is factored out. Smaller capacitors infer smaller devices and lower device costs. It also infers less charge shuttling between PC supply and the computational array, and in principle, reduced dissipative loss.

Practical considerations, such as in integration, TL design and C_{min} , will limit capacitive scaling. As such, the C_T ACN design choice is important. Picking the same value for C_T for all ACNs is one option. However, this is not considering the distribution of AN weight values, where small-valued weights can lead to mapped capacitance values below C_{min} . Trivial weight pruning is a possible solution, but destructive. Selecting each ACN C_T such that $\min(|w_i|)$ maps to C_{min} is another option, but this can lead to large C_T due to large AN weight ratios i.e. $\max(|w_i|)/\min(|w_i|)$. Regularization, use of quantizers with a dead zone, or pruning during ANN weight training are alternate solutions.

G. Capacitive arrays

ACN capacitors will likely form IC capacitive arrays during chip layout, not dissimilar to those used in Successive-Approximation Register (SAR) ADC designs [36]. Consequently, they face similar challenges, such as device mismatch in production [34], [37]. Unlike SAR ADCs with regular-sized capacitance values, ACNs can generate a wide distribution of capacitances. This is an extra challenge for IC designers, with more complex layouts and routing required.

Mapped ballast capacitors are specifically problematic, as their values vary significantly with the direction of \mathbf{w} and in proportion to C_T . Use of ballast pillars for v_m swing control can further increase C_d size. Implementation of large individual capacitors can come with their own practical issues and effects during IC manufacturing and during operation.

Building arrays using tiled, composite, equally-sized capacitors is one design approach [38]. If quantization during ANN training is used, more regularly-shaped arrays can be formed with less quantization errors. A mapped BNN with $w_i \in \{-1, +1\}$, is illustrated in Fig. 9. The symmetry property of a DTSC ACN is clearly shown. The downside of increasing levels of quantization during training is that potentially more ANs will be required to achieve the same level of functionality, compared to ANNs with real-valued weights.

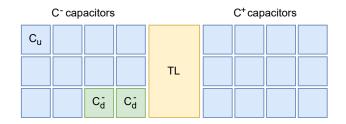


Fig. 9. Illustrative N=22 DTSC ACN capacitive array IC layout with $N^+=12$ and $N^-=10$ with equally-sized synapse unit capacitors, $C_u=C_T/22$ and conditionally-mapped ballast capacitors $C_d^-=(N^+-N^-)C_u$.

V. Software

To verify various ACN mappings, $10,000\ N=8$ real-valued weight vectors were randomly selected using a $\mathcal{N}(0,0.01)$ normal distribution; allowing for small w_T required for the ReLU mapping. Each vector was mapped and then tested with all possible 256 inputs with any errors between AN and ACN

TABLE I

10,0000 Random $\mathcal{N}(0,0.01)$ N=8 weight vector mapping testing with $C_T=100fF$, and capacitive pillars (if used) $C_{pb}=2fF,C_{pd}=5fF$, over all possible 256 input patterns

Mapping configuration	Mean C_d	Mean C
	(Dev.), fF	(Dev.), fF
Conditional, $\tau = 0.0$	36 (25)	0.66 (0.12)
Conditional vectored-bias, $\tau = 0.1$	32 (23)	0.62 (0.11)
Conditional, $\tau = 0.1$	52 (26)	0.56 (0.08)
Conditional, $\tau = 0.1$, and pillars	62 (26)	0.52 (0.07)
Conditional, $\tau = -0.1$	52 (26)	0.56 (0.08)
Balanced, $\tau = 0.1$	117 (5)	0.40 (0.03)
ReLU, $\tau = 0.1, V_{max} = 1.0V$	187 (72)	0.29 (0.06)

TABLE II $10,\!0000~{\rm Random}~\mathcal{N}(0,0.01)~{\rm conditional}~{\rm weight}~{\rm vector}~{\rm mapping}~~(\tau=0.0)~{\rm testing}~{\rm for}~{\rm different}~N~{\rm with}~C_T=100fF~{\rm against}~~{\rm equivalent}~{\rm binarized}~\{-1,1\}~{\rm version}$

	Real-valued weights		Binarized weights	
N	Mean C_d	Mean C	Mean C_d	Mean C
	(Dev.), fF	(Dev.), fF	(Dev.), fF	(Dev.), fF
8	36 (25)	0.66 (0.12)	27 (22)	0.57 (0.10)
16	25 (18)	0.50 (0.07)	20 (15)	0.42 (0.05)
32	18 (13)	0.38 (0.04)	14 (11)	0.31 (0.03)
64	13 (9)	0.28 (0.02)	10 (8)	0.23 (0.01)
784	4 (3)	0.09 (0.00)	3 (2)	0.07 (0.00)

outputs recorded. Table I provides results for 7 different mapping configurations. All software weights mapped correctly across all configurations, except for 12 errors apportioned to numeric rounding. The ReLU mapping had 22 weight vectors with excessive w_T that failed to map. As predicted, conditional mapping provided significantly better results than other options with minimal $C_d = \sum C_d^{\pm}$, maximal $|\mathbf{C}|$ and, consequently, Δv_m . Capacitive pillars were shown to work, with obvious extra capacitive cost and lower $|\mathbf{C}|$.

Results for conditional mapping and varying N are given in Table II. In this synthetic test, real-valued weights were also binarized to $\{-1,1\}$ for comparison. Results show that the average $|\mathbf{C}|$ decreased with increasing N, as expected. In this case binary weights, as predicted, has also led to smaller $|\mathbf{C}|$, even though they required smaller C_d . Mean C_d also reduces with increasing N. However, weights were drawn from a symmetric distribution, where $mean |w_T^+ - w_T^-|/w_T$ will naturally tend towards zero with increasing N.

Consequently, we explore the conditional-mapping behavior when using trained ANN weights that embed information in their distributions. Three different 2-layer ANN configurations were trained in Python 3.11.2 using a combination of Tensor-Flow (version 2.12.0) and the Larq (version 0.13.0) extension for QNNs and BNNs. Networks were trained on a binarized version of the 28x28 pixel MNIST image classification dataset. The MNIST binarization used binary '1' to represent pixel values above 127, binary '0' otherwise. The networks were trained with an Adam optimizer for 50 epochs using 60,000 training samples and 10,000 validation samples. Each network was trained five times with different initialized weights.

All trained ANNs had N=784 1-bit inputs and 10 linear output neurons with real-valued weights. ANs in the hidden layer used an ACN-compatible binary activation func-

TABLE III
MNIST EXPERIMENTAL NETWORK CONFIGURATIONS WITH
ACN-COMPATIBLE 784-BIT INPUT/1-BIT OUTPUT HIDDEN LAYER ANS

Name	Network configuration		
part32	ANN with 32 hidden-layer neurons with real-valued weights and biases.		
kbit36	QNN with 36 hidden-layer neurons with real-valued bias and 4-bit quantized weights with a [1,-1] clipping constraint.		
bin156	BNN with 156 hidden-layer neurons with binary $\{-1,+1\}$ weights and fixed bias, $\tau = 0.5$.		

tion and weights quantized differently per configuration. This demonstrated support for different ANNs, but also the impact of quantization on ACN properties, such as overall chip size and accuracy. The number of hidden layer ANs was selected to achieve similar classification performance across configurations. ANN configurations are provided in Table III. QNN/BNN networks *kbit36* and *bin156* used the Larq quantizers **DoReFaQuantizer** and **SteSign** respectively [30].

All hidden AN weights and biases were extracted from the trained models and conditionally mapped with constraints $C_{min}=2fF$ and $V_{max}=1.0V$. To ensure mapped ballast and bias capacitors are realizable ($\geq C_{min}$), bias and ballast capacitive pillars of 2fF were used.

Mapping part32 ANNs created an issue due to some weights being very small-valued, implying a very large C_T to ensure all C_i were realizable. Therefore, trivial weight pruning of $|w_i|$ below a threshold was used. This meant finding a balance between loss of classification performance and minimizing C_T . A $C_T=4704fF$ was chosen with a pruning threshold of 0.078, which led to an average 1% drop in overall classification but resulted in all ACN capacitors being realizable. The pruning process removed, on average, 44% of the weights.

The advantages of ANN quantization are now obvious. Quantization removes extremely small-valued and unrealizable capacitance values. The minimum $|w_i|$ is now determined by the quantizer. Setting $C_T = C_{min}w_T/min(|w_i|)$ for each N assures all C_i will be realizable. For kbit32 ANNs this resulted in an average $C_T = 4976fF$ with values ranging from 2198fF up to 6597fF. The network now has ACNs with significant variations in size. These irregularly-sized ACNs are now adding a new IC layout challenge. Note, bias values are not quantized in Larq **QuantDense** layers, which can lead to very small-valued trained biases. As such, using a vectored-bias mapping resulted in C_T values up to 35,240fF.

For the bin156 ANNs $C_i^\pm=C_{min}$ and $C_T=1568fF$. The bias capacitor is calculated as $C_b^-=\tau C_{min}=1fF$. This "half-C" bias prevents $\Delta v_m=0$ regardless of input. As $C_b^-< C_{min}$, the 2fF bias and ballast pillar applied to each tree maintains this functionality.

Binary-MNIST classification accuracy results are shown in Table IV. The TensorFlow ANN classification over 10,000 images was compared against the ANN inference when the hidden layer was replaced by the ACN mapped hardware model. The 1% drop in accuracy with part32 was expected due to pruning. The instability metric shows for the part32 ANN nearly 8% of computations would require the physical comparator circuit to resolve Δv_m below 5mV. Quantization

TABLE IV MNIST RESULTS WITH VALIDATION CLASSIFICATION ACCURACY FOR BOTH ORIGINAL SOFTWARE AND MAPPED ACN HARDWARE MODEL IN HIDDEN LAYER AND 5mV Instability Metric Ψ

Name	Mean Software (Dev.) (%)	Mean Hardware (Dev.) (%)	Instability Metric, Ψ
part32	92.0 (0.2)	91.0 (0.2)	0.08
kbit36	92.6 (0.2)	92.6 (0.2)	0.02
bin156	92.1 (0.4)	92.1 (0.4)	0.01

TABLE V
MNIST HIDDEN LAYER MAPPED ACN CAPACITANCE VALUE SIZES

Name	Mean C_T	Mean C_d	Mean C	Mean Layer
	(Dev.), fF	(Dev.), fF	(Dev.), fF	Size (Dev.), pF
part32	4704 (0)	1338 (101)	0.10 (0.00)	194 (3)
kbit36	4976 (114)	1718 (48)	0.08 (0.00)	242 (5)
bin156	1568 (0)	367 (8)	0.06 (0.00)	303 (1)

has significantly reduced instability, meaning fewer computations are potentially affected by comparator inaccuracies.

Table V shows the capacitance values for the mapped ACNs. The mean size of the ACN hidden layer increased with greater quantization and hidden neurons. However, bin156 was only roughly 50% greater than part32 due to the smaller C_T used, even though there were nearly 5 times the number of ANs.

Table VI gives $|\Delta v_m|$ statistics over all the test images and ACNs. As predicted by $|\mathbf{C}|$ in Table V, the maximum $|\Delta v_m|$ reduces with increased levels of quantization. However, the average part32 $|\Delta v_m|$ is half that of the quantized ANNs. From (22) this infers a difference in $\cos\theta$, as the inputs are common between tests. Fig. 10 shows the distribution of $\cos\theta$ over all tests for the part32 and bin156 networks. The direction of trained ANN weight vectors again has a significant effect on the characteristics of physical ACNs. Increased correlation of weight vectors with inputs in bin156 with the larger number of ANs is reducing potential comparator inaccuracies with, on average, larger $|\cos\theta|$ and $|\Delta v_m|$.

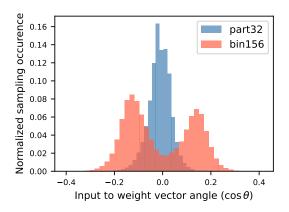


Fig. 10. Sampling distribution of $\cos \theta$ over all 10,000 validation samples for both the *part32* and *bin156* networks.

VI. DISCUSSION

The energy-efficient DTSC ACN uses an SC network to perform computation using charge provided by a time-varying

TABLE VI MNIST Absolute Differential Membrane Voltages Over 10,000 Validation Images

Name	Mean $ \Delta v_m $ (Dev.) (mV)	$ \begin{array}{c c} \operatorname{Max} \Delta v_m \\ (mV) \end{array} $
part32	39 (0.04)	481
kbit36	75 (0.06)	431
bin156	75 (0.04)	337

PC supply. From the PC perspective, the ACN network can be modeled as an RC load circuit comprising of a dynamic parasitic resistive load (R_L) and a dynamic switched capacitive load (C_L) . The dissipative energy, and consequently PC efficiency, is a function including these two variables [25]. The C_L will be a function of input but also mapping design choices such as C_T , the mapping algorithm and ANN weight vectors. Minimizing the ACN capacitance values, through an optimal mapping, is not only important for reducing both the chip area size and potential frequency fluctuations in the PC supply but also energy consumption.

The paper has provided a rigorous mathematical introduction for an IC designer on mapping ANs to ACNs, as well as practical tools and pointers. DTSC ACN requires only Nsynapse capacitors compared to 2N solutions [15] and the mapping provides an exact value for C_d . We have shown how metrics, such as Ψ and $|\Delta v_m|$ expose the demands on TL performance. The introduction of SPDT switches in DTSC also provides a linear relationship between Δv_m and the input, like the ANN $\mathbf{w} \cdot \mathbf{x}$ dot product. Finally, the ACN allows IC designers to build arbitrary-sized capacitors to represent arbitrary-precision ANN weights. However, as we have seen, there are also significant advantages to weight/capacitance value quantization. In the extreme, binary weighted, regularlyshaped, ACNs will look attractive to IC designers familiar with ADC arrays. Smaller, binarized ACN designs, crucially reduce capacitor design burden but will need to consider the energy trade-off with an increase in the number of ACNs required to achieve equivalent levels of functionality.

Going forward, thoughts turn to how large, multiple ACN, multi-layer networks will be physically constructed and their associated cost. Mismatch IC issues may be problematic, as well as input routing, PC distribution and ability to accurately instantiate required capacitance values. Networks will need to tackle increasingly demanding applications. More generic, reusable, weight configurable and re-programmable ACNs will be required. Thought should also be given to more ACNfriendly software training tools. Given knowledge of the mapping properties, thought can now be applied to ACN-specific tools, such as quantizers, regularizers and pruning techniques, that promote smaller values for C_T , or punish weights that unnecessarily promote large mapped ballast capacitance values or small Δv_m . Consideration will also be required with respect to more complex networks, such as the ubiquitous Convolutional Neural Network (CNN). Support for multi-bit output ACNs and popular activation functions, such as ReLU, will need to be investigated in a drive towards increasingly efficient and cost effective computation. The present work, thus, lays a key foundation stone in a wider field of study

where we see significant potential for innovation.

VII. CONCLUSION

This paper has demonstrated that ANNs trained using standard tools, such as TensorFlow, with real-valued, quantized or binary signed weights, can be mapped optimally and robustly onto energy-efficient, adiabatic, DTSC ACNs. Specifically, we have shown that choosing DTSC ACNs as the fundamental adiabatic neuron unit implementation and combining it with the conditional mapping approach we can minimize total capacitance and chip area, and maximize the differential voltages applied to physical, non-ideal comparators. We have seen how weight quantization can play a significant role in generating practical and realizable designs. Lastly, properties like symmetry, voltage and capacitive scaling, as well as tools such as capacitive pillars, make DTSC ACNs an attractive solution for designers looking for energy-efficient solutions.

Although challenges remain, the future of adiabatic logic and charge recovery appears ever more promising for the design and implementation of large ANNs, as well as other computationally intensive applications. Furthermore, the regularity and algorithmic nature of the mapping process holds tantalizing prospects for future automation of the design of DTSC ACN networks.

REFERENCES

- [1] P. Teichmann, "Adiabatic logic: Future trend and system level perspective," *Springer Series in Advanced Microelectronics*, vol. 34, 2012.
- [2] W. C. Athas, J. G. Koller, and L. J. Svensson, "Energy-efficient cmos line driver using adiabatic switching," in *Proceedings of the IEEE Great Lakes Symposium on VLSI*, 1994.
- [3] J. G. Koller and W. C. Athas, "Adiabatic switching, low energy computing, and the physics of storing and erasing information," in *Proceedings of the Workshop on Physics and Computation, PhysComp* 1992, 1992.
- [4] D. Maksimović, V. G. Oklobdžija, B. Nikolić, and K. W. Current, "Clocked cmos adiabatic logic with integrated single-phase power-clock supply," *IEEE Trans on VLSI Systems*, vol. 8, 2000.
- [5] S. Maheshwari, A. Serb, C. Papavassiliou, and T. Prodromakis, "An adiabatic capacitive artificial neuron with rram-based threshold detection for energy-efficient neuromorphic computing," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, pp. 3512–3525, 9 2022.
- [6] S. Nakata et al., "General stability of stepwise waveform of an adiabatic charge recycling circuit with any circuit topology," *IEEE Trans. on Circuits and Systems I: Regular Papers*, vol. 59, pp. 2301–2314, 2012.
- [7] H. S. Raghav, V. A. Bartlett, and I. Kale, "Investigation of stepwise charging circuits for power-clock generation in adiabatic logic," in 12th Int Conf on PhD Res in Microelec and Elec (PRIME), 2016, pp. 1–4.
- [8] —, "Energy efficiency of 2-step charging power-clock for adiabatic logic," in *Proceedings - 26th International Workshop on Power and Timing Modeling, Optimization and Simulation*, 2016.
- [9] H. Thapliyal and S. D. Kumar, "Adiabatic logic-in-memory architecture," p. 761, 12 2020.
- [10] M.-C. Chang and Y.-T. Kuo, "Design of a two-phase adiabatic contentaddressable memory," Tech. Rep., 2011.
- [11] R. Lal, W. Athas, and L. Svensson, "A low-power adiabatic driver system for amleds," in 2000 Symposium on VLSI Circuits Digest of Technical Papers, 2000, pp. 198–201.
- [12] H. S. Raghav and I. Kale, "A balanced power analysis attack resilient adiabatic logic using single charge sharing transistor," *Integration*, vol. 69, pp. 147–160, 11 2019.
- [13] S. Maheshwari, V. A. Bartlett, and I. Kale, "Energy efficient implementation of multi-phase quasi-adiabatic cyclic redundancy check in near field communication," *Integration*, vol. 62, pp. 341–352, 6 2018.
- [14] M. Massarotto, S. Saggini, M. Loghi, and D. Esseni, "Adiabatic spiking neurons and synapses for ultra-low energy neuromorphic computing," in 30th IEEE Int Conf on Electronics, Circuits and Systems, 2023.

- [15] ——, "Adiabatic leaky integrate and fire neurons with refractory period for ultra low energy neuromorphic computing," npj Unconventional Computing, vol. 1, p. 15, 12 2024. [Online]. Available: https://www.nature.com/articles/s44335-024-00013-1
- [16] B. J. Maundy and E. I. El-Masry, "A self-organizing switched-capacitor neural network," *IEEE Trans on Circuits and Systems*, vol. 38, 1991.
- [17] D. Bankman and B. Murmann, "An 8-bit, 16 input, 3.2 pj/op switched-capacitor dot product circuit in 28-nm fdsoi cmos," in 2016 IEEE Asian Solid-State Circuits Conference, A-SSCC 2016 Proceedings, 2017.
- [18] S. D. Giacomo et al., "A cmos implementation of a switched capacitor analog neural network asic," in IEEE Nuclear Science Symp, Medical Imaging Conf and Int Symp on Room-Temperature Semiconductor Detectors, 2023.
- [19] D. Hajtas et al., "Switched capacitor-based integrate-and-fire neural network," in The State of the Art in Computational Intelligence. Advances in Soft Computing. Springer, 2000, pp. 50–55.
 [20] W. Zhu, H. Yin, Y. Zhao, G. Yu, Y. Yang, and C. Wang, "A high-
- [20] W. Zhu, H. Yin, Y. Zhao, G. Yu, Y. Yang, and C. Wang, "A high-linearity, energy-efficient switched-capacitor computing circuit for edge applications," in *Proceedings IEEE International Conference on Integrated Circuits, Technologies and Applications*, 2023, pp. 103–104.
- [21] Y. Tsividis and D. Anastassiou, "Switched-capacitor neural networks," Electronics Letters, vol. 23, pp. 958–959, 8 1987.
- [22] M. Ronchi et al., "Design, implementation, and analysis of an integrated switched capacitor analog neuron for edge computing ai accelerators," IEEE Trans. on Circuits and Systems I: Regular Papers, 2025.
- [23] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, "A 64-tile 2.4-mb in-memory-computing cnn accelerator employing charge-domain compute," *IEEE Journal of Solid-State Circuits*, vol. 54, 2019.
- [24] S. Maheshwari, A. Serb, C. Papavassiliou, and T. Prodromakis, "An adiabatic regenerative capacitive artificial neuron," in *Proceedings -IEEE Int Symp on Circuits and Systems*, 2021.
- [25] S. Maheshwari, M. Smart, H. S. Raghav, T. Prodromakis, and A. Serb, "Adiabatic capacitive neuron: An energy-efficient functional unit for artificial neural networks," 7 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2507.00831
- [26] I. Hubara et al., "Quantized neural networks: Training neural networks with low precision weights and activations," *Journal of Machine Learn*ing Research, vol. 18, 2018.
- [27] H. Qin, R. Gong, X. Liu, X. Bai, J. Song, and N. Sebe, "Binary neural networks: A survey," *Pattern Recognition*, vol. 105, 2020.
- [28] C. Yuan and S. S. Agaian, "A comprehensive review of binary neural network," *Artificial Intelligence Review*, 2023.
- [29] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," 2013.
- [30] L. Geiger and P. Team, "Larq: An open-source library for training binarized neural networks," *Journal of Open Source Software*, vol. 5, p. 1746, 1 2020.
- [31] M. P. H. Lin et al., "Common-centroid capacitor layout generation considering device matching and parasitic minimization," *IEEE Trans*actions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, pp. 991–1002, 2013.
- [32] C.-W. Lin et al., "Common-centroid capacitor placement considering systematic and random mismatches in analog integrated circuits," in Proceedings of the 48th Design Automation Conference. ACM Digital Library, 2013, pp. 528–533.
- [33] A. Abusleme, A. Dragone, G. Haller, and B. Murmann, "Mismatch of lateral field metal-oxide-metal capacitors in 180 nm cmos process," *Electronics Letters*, vol. 48, pp. 286–287, 3 2012.
- [34] T. Wakimoto, H. Li, and K. Murase, "Statistical analysis on the effect of capacitance mismatch in a high-resolution successive-approximation adc," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 6, pp. 89–93, 2011.
- [35] L. T. Wang, "Process variation in metal-oxide-metal (mom) capacitors," in *Design for Manufacturability through Design-Process Integration II*, vol. 6925. SPIE, 3 2008, p. 69251M.
- [36] A. Ahuja et al., "Comparison of capacitive dac architectures for power and area efficient sar adc designs," in Proceedings - IEEE International Symposium on Circuits and Systems, vol. 2021-May, 2021.
- [37] R. Fiorelli et al., "Effects of capacitors non-idealities in un-even split-capacitor array sar adcs," in 2015 Conference on Design of Circuits and Integrated Systems, DCIS 2015. IEEE, 2016.
- [38] N. C. Chen et al., "High-density mom capacitor array with novel mortise-tenon structure for low-power sar adc," in Proceedings of the 2017 Design, Automation and Test in Europe, DATE 2017, 2017.

VIII. BIOGRAPHY SECTION



Mike Smart received the degree of electrical engineering from the University of Warwick, U.K. and the Ph.D. degree in electrical engineering from the University of Edinburgh, U.K., in 1992 and 1996 respectively. He has worked as a staff engineer for Motorola Solutions and a lead engineer for Indigo Vision Ltd. He is currently a Software Engineer at the University of Edinburgh, U.K. His research interests include AI, novel computation, algorithms, video compression and biologically-inspired software.



Sachin Maheshwari (S'12–M'21) received a B.E. in Electrical and Electronic Engineering from ICFAI University, an M.E. in Microelectronics from BITS Pilani, and a Ph.D. in Electronics Engineering from the University of Westminster, UK. He was a Research Fellow at the University of Southampton and is currently a Research Associate at the Centre for Electronics Frontiers, University of Edinburgh. His research focuses on neuromorphic computing and artificial neural networks, with emphasis on energy-recovery logic (adiabatic techniques) and emerging

technologies (RRAM) for energy-efficient brain-inspired systems.



Himadri Singh Raghav received her B.Sc. and M.Sc. in electronics and M.Tech. in VLSI Design from Banasthali University, Rajasthan, India. She then obtained her Ph.D. in Electronics Engineering from the University of Westminster, London, UK. She worked for 3 years as a Research Fellow at the National University of Singapore. She is currently working as a Research Associate with the Centre for Electronics Frontiers, School of Engineering, University of Edinburgh, UK. Her research interest is in energy efficient implementation of a secure

system using charge recovery logic



Alexander Serb received his degree in Biomedical Engineering from Imperial College in 2009 and his PhD in Electrical and Electronics Engineering from Imperial College in 2013. Currently, he is a reader at the University of Edinburgh, UK. His research interests are: cognitive computing, neuroinspired engineering, algorithms and applications using RRAM, RRAM device modelling and instrumentation design.