Bias-Scalable Near-Memory CMOS Analog Processor for Machine Learning

Pratik Kumar, Student Member, IEEE, Ankita Nandi, Student Member, IEEE, Shantanu Chakrabartty, Senior Member, IEEE, and Chetan Singh Thakur, Senior Member, IEEE

Abstract—Bias-scalable analog computing is attractive for implementing machine learning (ML) processors with distinct power-performance specifications. For instance, ML implementations for server workloads are focused on higher computational throughput for faster training, whereas ML implementations for edge devices are focused on energy-efficient inference. In this paper, we demonstrate the implementation of bias-scalable and temperature-scalable analog computing circuits using the generalization of the margin-propagation principle called shape-based analog computing (S-AC). The resulting S-AC core integrates several near-memory compute elements, which include: (a) non-linear activation functions; (b) inner-product compute circuits; and (c) a mixed-signal compressive memory, all of which can be scaled for performance or power while preserving its functionality. Using measured results from prototypes fabricated in a 180nm CMOS process, we demonstrate that the performance of computing modules remains robust to transistor biasing and variations in temperature. In this paper, we also demonstrate the effect of bias-scalability and computational accuracy on a simple ML regression task.

Index Terms—Analog approximate computing, generalized margin-propagation, shape-based analog computing, machine learning, memory DAC, analog multiplier, ReLU.

I. INTRODUCTION

Analog computing offers a novel paradigm for designing machine learning (ML) systems [1], [2], [3], [4], [5] because the circuits can exploit computational primitives inherent in the device physics along with conservation principles to achieve very high computational density and energy efficiency. However, conventional analog computing circuits operate within a pre-defined transistor biasing regime (weak-inversion [6] or strong-inversion [7]) to ensure sufficient dynamic range and compliance with respect to temperature variations. This approach limits scaling the design across applications that demand distinct power-performance specifications. An illustration of this trade-off is highlighted in Fig. 1a. It can be observed from Fig. 1a that when the transistors are biased in strong-inversion (SI), higher speed can be achieved but at the cost of increased power consumption, signifying a lower TOPS/W (Trillions Operations per Second per Watt). In weak-inversion (WI), higher energy efficiency (or higher TOPS/W) can be achieved but at the expense of lower speed. Irrespective of the biasing conditions, it is desirable that the functionality of the analog compute and memory circuits remain invariant. This property is called bias-scalability [8]. Bias-scalable analog circuits were first reported in [9] and used a margin-propagation (MP) principle [10]. In [11], the MP principle was generalized to shape-based computing, which endowed the analog computing circuits to be process-scalable and temperature-scalable. This work proposes a near-memory shape-based analog computing (S-AC) processing core that combines S-AC multipliers, S-AC non-linear activation, and S-AC compressive digital-to-analog converter (DAC), which are integrated in proximity with each other and are important for ML applications. We have exploited the bias and temperature scalability of the MP principle to achieve faster training (using above threshold biasing) and achieve energy-efficient inference (using sub-threshold biasing) without resorting to any post-training calibration. All the results reported in this paper have been measured from a fabricated prototype, unlike our previous results reported in [11] which were based on circuit simulation of basic computational units.

Manuscript received 27 September 2022; revised 6 December 2022; accepted 3 January 2023. Date of publication 5 January 2023; date of current version 20 March 2023. This work was supported by the Department of Science and Technology of India, Science and Engineering Research Board (SERB), under Grant CRG/2021/005478 and Grant DST/IM/P/2018/000550. This article was recommended by Guest Editor P. Sotiriadis. (Corresponding author: Pratik Kumar)

Pratik Kumar, Ankita Nandi, and Chetan Singh Thakur are with the NeuRonICS Laboratory, Department of Electronic Systems Engineering, Indian Institute of Science, Bengaluru 560012, India (e-mail: csthakur@iisc.ac.in).

Shantanu Chakrabartty is with the Department of Electrical and Systems Engineering, Washington University in St. Louis, St. Louis, MO 63130 USA. Color versions of one or more figures in this article are available at https://doi.org/10.1109/JETCAS.2023.3234570.

Digital Object Identifier 10.1109/JETCAS.2023.3234570.

Fig. 1. (a) Desired performance efficiency (TOPS/W) and operational speed plot for a system that can be tuned to execute both edge application and server workload; (b) S-AC based near-memory ML core depicting basic computational unit such as Multiplier, DAC, ReLU, Adder along with local memory elements (for storing trained parameters) required for efficient ML computation.
As a proof of concept, this paper also presents the standard regression task at variable speed and power consumption.

The key contributions of this work in relation to previous approaches are as follows:

- Design proposition of S-AC based near-memory analog compute core as depicted in Fig. 1 which utilizes S-AC multiplier, S-AC non-linearity, and S-AC log-compressive memory digital-to-analog converter (DAC) at its core.
- Design proposition of digitally programmable compressive memory DAC utilized as a transformation block near the processing element and at the interface to the external world. This enables scalable memory elements along with scalable analog ML core for faster system adaptation.
- Design implementation of S-AC based four quadrant approximate multiplier.
- Design validation of proposed S-AC core to implement a standard ML regression task. This results in a near-memory S-AC compute paradigm where the designed S-AC analog ML system can be scaled both for performance or energy. We also verify the functional capabilities of the S-AC system to operate at different computational accuracy and power while maintaining the overall system’s performance.

The rest of the sections are organized as follows. Section II describes the background of S-AC circuits and GMP formulation. Section III shows the design of basic computational circuits truly scalable across different bias conditions. This makes S-AC circuits truly scalable across different bias conditions.

Now for a given input matrix $X \in \mathbb{R}^{N \times S}$ where $x_i \in \mathbb{R}^N$, $\forall i = 1, \ldots, N$ is the input vector and $x_j \in \mathbb{R}^S$, $\forall j = 1, \ldots, S$ is the number of splines implementing the approximation, the CMOS circuit satisfying constraints in (1) and (2) is shown in Fig. 2a. Here, $x_i,j$ is the input current for the $i^{th}$ input and the $j^{th}$ spline, $h(X) = f(V_B, 0)$ is the output current, $V_{i,j}$ and $V_B$ are the voltages across the $N_i,j$ transistor, $C$ is a constant current, and $D_{i,j}$ denotes diode elements (Schottky, MOS diode or any other). Applying KCL at node $V_B$, (1) can be obtained while the current across diode $D_{i,j}$ gives (2).

It can further be noted that the current source $C$ implements a constraint that dictates all the diode currents to add up to $C$ where the diode element itself forces the flow of current in one direction, similar to rectification operation. In addition, transistors $N_{i,j}$, $\forall i = 1, \ldots, N$ and $j = 1, \ldots, S$ form the current mirror branches, thereby enforcing similar currents among all the mirror branches. Due to these simultaneous constraints imposed by the current source $C$, diode elements $D_{i,j}$, and the mirror branches $N_{i,j}$, the implemented circuit in Fig. 2a settles to a unique value of $V_B$ which satisfies all these constraints. This unique value of $V_B$ finally results in output $h$ i.e., $h(X) = f(V_B, 0)$ as the solution to (1) and (2). Furthermore it can be emphasized from [11] that the output $h(\cdot)$ (also called proto-shape) similar to function $f(\cdot)$ always satisfies the properties (3) and (4) given by

$$1 \geq \frac{\partial h}{\partial x_i} \geq 0, \forall i \quad (3)$$

$$\lim_{x_i \to \infty} \frac{\partial h}{\partial x_i} = 1$$

$$\lim_{x_i \to -\infty} \frac{\partial h}{\partial x_i} = 0 \quad (4)$$

The property in (3) ensures that the obtained proto-shape $h$ is monotonic with respect to its variable, while the properties described by (4) determine the two asymptotes of $h$, irrespective of the specific form of $f$. Fig. 2b shows the example of the obtained shape using the circuit in Fig. 2a for input dimension $N = 1$ and the design parameter $S = 3$. The results are also shown for different MOSFET biasing regimes, i.e., WI, MI, and SI biasing regimes which correspond to different functions $f(\cdot)$ in (1) and (2). This shape will often be referred to as the basic proto-shape in the following text. Fig. 2c shows that other complex non-linear monotonic shapes can further be constructed using the proto-shape by using only rotation and translation techniques.

III. NEAR-MEMORY S-AC CORE DESIGN

This section presents the building blocks required for designing a near-memory analog ML core. An adequate requirement for an ML inference processor incorporating
Fig. 2. Background on S-AC and MP circuits: (a) Implementation of N-type S-AC circuit [9], [11] for N inputs and S splines, the inset shows the circuit implementation of a single S-AC unit using n-type FET and a diode; (b) Basic S-AC shape (also called proto-shape) implemented by the N-type S-AC circuit when a single input x is varied for different operating regimes and design parameter S = 3; (c) Different non-linear functions (A1 − A4), (B1 − B2) that can be implemented by translation, rotation, addition and subtraction of the proto-shape shown in Fig. 2b.

Fig. 3. (a) Compressive log-binary DAC implementation using S-AC; (b) Comparison plot between S-AC log-binary DAC, Bfloat16 & IEEE32 number systems demonstrating close compliance of compressive nature of corresponding log2 curves with each other.

analog ML cores requires: (a) memory for storing the inference parameters and for supporting a digital interface for inputs; (b) multiply-accumulate circuits; and (c) non-linear computing circuits. Here we show that the basic S-AC circuit shown in Fig. 2a can be modified and extended to implement all the required computational building blocks. We specifically implement a combination of a compressive mixed-signal memory DAC and a non-linear multiplier circuit that results in a multiply-accumulate (MAC) operation, which also emulates computing using Bfloat16 number representation [15]. It may be noted that any approximation error introduced in this mapping can be compensated during training itself, as neural networks are resilient to error and can adapt to approximation errors if properly trained [16].

A. S-AC Based Compressive Memory DAC

One of the major challenges in implementing an analog ML processor is storing and updating trained parameters. While analog memories based on memristors, floating gates, and other nano-scale devices have been proposed for analog ML processors [17], [18], [19], their functional response and speed do not scale across training and inference. Therefore, in this paper, we propose to use a DAC-based memory that uses a S-AC based analog front-end to implement a compressive function. This compressive function will be utilized by the S-AC multiplier (discussed in Section III-B) to implement a memory-compute block for MAC operation. Here we show that this compressive-expansive operation is equivalent to analog computing using Bfloat16 [15] and the IEEE-754 single-precision (32-bit) number systems. Note that the Bfloat16 number system developed by Google Brain delivers more accurate results at lesser hardware as compared to IEEE 754 single-precision numbers for some neural networks and is extensively used by Google cloud TPUs [15]. Consider a function \( g(x) \) given by

\[
g(x) = \log_2 \left( \sum_i 2^{b_i} \right). \tag{5}
\]

Then, it is easy to verify that \( g(x) \) satisfies the properties

\[
1 \geq \left| \frac{\partial g}{\partial x_i} \right| \geq 0 \tag{6}
\]

\[
\lim_{x_i \to \infty} \frac{\partial g}{\partial x_i} = 1 \tag{7}
\]

similar to that of the proto-shape \( h(x) \) in (3) and (4). If \( x \) is denoted by its binary representation as \( x \approx \sum_{i=1}^{N} 2^{b_i} \), then
When S be rewritten for any base $\theta$, DAC. Using the base change property of logarithm, (8) can implicitly expanded in (10) for multiplication.

The proto-shape $h(x)$ can be used to implement analog multipliers based on the following Taylor series approximation

$$
\begin{align*}
&h(C + w + C - x) - h(C + w + C + x) \\
&\approx 2x \times \left( \frac{dh(C + w)}{dw} - \frac{dh(C - w)}{dw} \right) \\
&\approx 2x \times (w^2 - w^-)
\end{align*}
$$

The constant $C$ ensures that the input to the proto-shape $h$ is always positive. The differential combination effectively cancels the zeroth order and second-order terms in the Taylor series [16] and the property of $h$ in (3), leads to (10). The detailed derivation is provided in Appendix A. Note that one of the differential arguments to the multiplier $(w^2 - w^-)$ is a non-linear map $\frac{dh}{dw}$, which, based on property (3), is a compressive map. Thus, the stored parameters need to be pre-processed before and are presented as an input to the multiplier. This is the basis for our compressive memory design described in Section III-A.

The circuit in Fig. 4a implements the scalar multiplication given in (10) where $w \in \mathbb{R}$, $x \in \mathbb{R}$ and the product $y \in \mathbb{R}$. Fig. 4a shows the S-AC unit utilized to implement each component in (10). The inputs are first converted into their analog equivalent and constant ($C$) is added to the negative term to shift the operation in the first quadrant. The output from all S-AC3 (here subscript 3 represents 3—spline S-AC) units is added and subtracted (differentially) as per (10) to obtain the desired multiplication. Fig. 4b shows a close approximation between the simulated output of the four-quadrant multiplier and the output obtained from an ideal multiplier. Based on this basic operation, multiply-accumulate operations and inner products can now be implemented by combining element-wise S-AC multipliers with summing circuits based on Kirchhoff’s
C. S-AC Based ReLU Activation

A soft ReLU function can be implemented using a one-dimensional proto-shape shown in Fig. 2b. The circuit implementation of the soft ReLU function is shown in Fig. 5. The basic circuit uses two S-AC units, one of which receives an input \( x \), and the other is driven by a zero current (or floating). It may be noted that as limit \( C \to 0 \), the proto-shape converges to an ideal ReLU function. Other non-linear functions can also be implemented by shift, translation, and addition of the basic proto-shape, as illustrated in Fig. 2c.

IV. MEASUREMENT RESULTS

The S-AC building blocks, along with computational nodes of S-AC based neural network, have been prototyped in a standard CMOS 180nm process technology. Fig. 6a shows the die microphotograph of the chip where a copy of basic computational blocks and the S-AC node has been highlighted. It may be noted that multiple copies of basic computational blocks were fabricated for test purposes. The functionality of the circuit modules has been verified using the test measurement setup shown in Fig. 6a. The test chip was mounted on a custom IC test board, and the test vectors were generated using a PYNQ-Z2 FPGA board which used a python-based interface to control the digital inputs and outputs. High-precision analog test equipments were directly interfaced with the test chip and were controlled by the PYNQ-Z2 FPGA board. To accurately determine the region of operation, the transistors were first characterized for some fixed circuit parameters (such as the aspect ratio of the transistor, spline count \( S \), etc.). The inversion coefficient (IC) range [22] was then used to find an approximate one-to-one mapping between the IC range and the bias current range. For strong inversion regime of operation, circuits were biased so as to maintain the inversion coefficient i.e. \( IC > 10 \) while for weak inversion \( IC < 0.1 \) was maintained and all the way between \( 0.1 < IC < 10 \) was marked as moderate inversion [22].

A. S-AC Compressive Memory Measured Result

Fig. 7a shows the measured result of 8-bit S-AC based DAC as a function of equivalent decimal input varying from 0 to 255 at different operating regimes. It can be seen that the result closely approximates the desired ideal shape and the output shape is invariant across operating regimes. With the increase in constant current \( C \), along with the offsets \([C_{1,1}, C_{1,2}, C_{1,3}, \ldots, C_{N,3}]\) for \( S = 3 \), the S-AC based DAC operation moves from WI to SI resulting in increased power consumption but simultaneously reducing settling time and in turn improving throughput and speed. However, the optimum trade-off between energy and throughput can be obtained in the MI region of operation.

B. S-AC Multiplier Measured Results

Fig. 7b and Fig. 7c shows the measured result of the implemented S-AC multiplier circuit for different values of design parameter \( S \) and at different operating conditions. Fig. 7b shows the comparison plot of a four-quadrant multiplier for design parameters \( S = 1 \) and \( S = 3 \). It can be noted that with the increase in design parameter \( S \), the multiplier accuracy increases and becomes much closer to the ideal. Fig. 7c is computed for \( S = 3 \), and shows the four-quadrant multiplication at different operating regimes in close compliance with each other. The results of Fig. 7b and Fig. 7c have been computed for \( x \in (-1, 1) \) and for \( w = [-0.5, 0.5] \). Fig. 7d shows the multiplication curve for \( x \in (-2, 2) \) and for different values of \( w \).

C. S-AC ReLU Measured Results

Fig. 8 shows the measured results of S-AC based ReLU implementation (Fig. 5) and its comparison with the ideal. It can be observed that the obtained normalized output current curve follows the desired non-linear shape and matches the
ideal. Furthermore, the non-linear shape remains invariant in weak, moderate, and strong inversion regimes as desired. The average power consumption varies between 18.2 nW to 89.4 µW with an area consumption of 190.46 µm² in 180nm technology node when the circuit operation shifts from WI to MI, respectively. This is much lesser than the corresponding digital implementation where the power reported in [23] is of the order of 1 mW at 40nm technology node with an area consumption of 1697.37 µm².

V. DESIGN SPACE TRADE-OFFS ANALYSIS

A. S-AC Area and Power Saving Analysis

S-AC design offers a range of trade-offs between accuracy, area, and power benefits by changing the design parameter (S). The value of this design parameter S is determined based on the application requirements. Theoretically, the number of splines selected can vary from 1 to S, therefore offering a wide range of trade-offs to choose from. On analysis, it was found that each increment in design parameter S, decreases the approximation error exponentially at the cost of additional \( \approx 20\% \) area requirement as compared to previous S. The performance evaluation of the S-AC multiplier found that with an average absolute error of 3.66\%, \( S = 3 \) offers up to 31.3\% in area savings and up to 37.2\% in power savings.

B. Design Margin and Shape Analysis

The shape-based analog computing framework is designed to preserve its transfer function within a stringent error margin under different biasing conditions (WI, MI, and SI) and temperature variations. This, in turn, suggests that irrespective of mismatches between the drain-source currents or the gate-source voltages for a matched pair of MOS, the proto-shape (\( h(\cdot) \) in Fig. 2b) remains intact. This proto-shape is considered important in machine learning applications and is governed by the design parameter S, which also decides the ability of the system to replicate the desired functional shape closely. S-AC design also relaxes the bounds of precise computing by allowing the user to choose the proto-shape (by choosing design parameter S) as per the need of the application and focus on obtaining desired functional shapes rather than conventional design techniques. Furthermore, like digital designs, the S-AC designs allow the user to trade off computational precision (by varying S) with energy and area [11]. On analysis, it was found that the design parameter \( S = 3 \) is good enough to match most of the desired shapes with \( > 98\% \) accuracy, while even with the design parameter \( S = 1 \) and \( S = 2 \), the classification system accuracy does not drop significantly as the network learns with hardware approximation.

C. Energy Analysis

Table I shows the best-case average energy consumed by S-AC based basic operational units. It can be noted that for strong inversion regime of operation, circuits were biased so as to maintain the inversion coefficient, i.e., \( I_C > 10 \), while for weak inversion, \( I_C < 0.1 \) was maintained [22]. This range of bias current corresponding to a particular operating regime was fed into the circuit to control the operating regime of the S-AC unit. It can be seen that as the circuit operating regimes move from SI to WI, the energy per operation decreases, while...

---

**Table I**

| @ S=1, VDD= 1.1V | # S-AC Unit | Energy Consumption (pJ) |
|------------------|-------------|------------------------|
| S-AC             | 1           | 1.49       | 0.34 |
| Multiply/Divide  | 4           | 5.23       | 1.19 |
| Soft-ReLU        | 2           | 2.51       | 0.59 |

*For operation in Strong Inversion (SI) regime Inversion Coefficient; \( I_C > 10 \) & for Weak Inversion (WI) regime \( I_C < 0.1 \) [22].
an optimal balance between power and speed will always be obtained in the MI regime.

D. Performance Analysis

The most significant errors introduced in the operation of S-AC circuits are represented by mismatches, noise, and power-supply variations. As a result of these undesired effects, the functionality of the circuits can be severely affected by additive errors. In S-AC circuits, the margin between the shapes obtained in the SI and WI regimes takes into account all the variations due to second-order effects. This crucial feature allows the S-AC circuits to preserve the inherent shape of the implemented function.

1) Temperature Variation: We compare the effect of nominal temperature variation on S-AC units. Fig. 9 shows the measured characteristic curves of S-AC based ReLU, Multiplier, and DAC at different temperature points, respectively. One can observe that even though there is a slight variation that can be attributed to the current mirrors in the desired curves, the overall characteristic shape is preserved.

2) Power & Task-Energy Efficiency: Fig. 10a shows a comparison plot between the measured and simulated power of S-AC based unit when the operating current is varied such that circuit operations move from WI to SI regime. It can be observed that the power consumption increases when circuit operation shifts from WI to SI regime.

3) Slew Rate: With the increase in the number of S-AC blocks, the corresponding slew rate and bandwidth increase as the number of inputs and the overall current available to charge the node capacitance increases. This results in an overall reduction in settling time and can be solely attributed to the constraints imposed by the hyper-parameter $C$ in (1). It can also be noted that as the value of this hyper-parameter $C$ decreases, i.e., when the circuit operation shifts from SI to the WI regime, the settling time increases because it takes more time for the capacitor at the gate of the output transistor (node $V_B$ in Fig. 2a) to charge with the limited available current.

4) Settling Time: This settling time (including dead time, slew time, and recovery time) decides the maximum input frequency at which the system can operate (assuming all the operations to be performed are done parallel) and can be given by (11)

$$f_{\text{max}} = \frac{1}{\text{max} (t_{\text{settling, rise}}, t_{\text{settling, fall}}) + \Delta t}$$  \hspace{1cm} (11)$$

Here, $\Delta t$ is the margin for the unexpected error that can arise due to circuit variations [24]. It can safely be assumed to be between 5% of $t_{\text{settling}}$. Fig. 10b shows the measured
Fig. 11. (a) System-level architecture of a 3-layer neural network, the inset shows the implementation of S-AC based near-memory core shown in Fig. 1 and implemented using S-AC DAC (Fig. 3a), S-AC multiplier (Fig. 4a), S-AC ReLU (Fig. 5) and S-AC WTA [11]: Measurement results of a Sine regression task performed on a 3-layer neural network showing: (b) training curve obtained in SI region and its comparison with ideal; (c) inference curve obtained in MI and WI region and its comparison with ideal; (d) Inference curve obtained for different design parameter $S$ in SI region and its comparison with ideal.

settling time of an S-AC based unit when the operating current is varied such that the circuit moves from WI to the SI region of operation. It can be observed that as the operating regime moves from WI to SI, the time required to charge the capacitance node improves. Hence the circuit can operate at a higher speed. Fig. 10c shows the variational performance efficiency ($P_E$) and system efficiency ($S_E$) when the circuit operating regime shifts from WI to SI. Note that $P_E$ increases with the increase in operating current while $S_E$ deteriorates as predicted in Fig. 1a.

VI. REGRESSION RESULTS

In this section, we demonstrate the functionality of the S-AC based 3-layer neural network on a simple regression task. Fig. 11a shows the neural architecture of a S-AC based 3-layer neural network containing 6 hidden nodes and its corresponding circuit implementation. Here each node implements an analog ML core as shown in Fig. 1b. Here, S-AC compressive memory units are used to store weights in the compressed log domain. This near-memory computing architecture reduces the energy wasted in moving data to and from the memory while simultaneously can operate at different bias currents. The parallel-connected S-AC multipliers whose outputs converge into a single node is representative of the proposed S-AC based Multiply-and-Accumulate (MAC) operation. However, the explanation and detailed implementation of S-AC based MAC and training methodology is beyond the scope of this work and shall be surfaced in the upcoming literature. The inputs to the architecture are first converted into differential compressive form and then passed to the hidden nodes. For demonstration, we use the S-AC architecture to learn a two-dimensional non-linear function given by

$$Y = \sin (2\pi x_1) \sin (2\pi x_2)$$

The network was trained keeping in account the device mismatch obtained from post-layout simulation. Fig. 11b shows the training curve obtained in the SI regime for $S = 1$ and its comparison with the ideal curve. We further show in Fig. 11b that when the same architecture is used for testing while operating in the WI or MI regime, we are able to achieve nearly similar plots. Thus, we show how bias scalability can be used as an advantage to perform high-speed training in SI, whereas testing is done for low-power in WI using the same hardware. Fig. 11d shows the Sine regression curve obtained for varying design parameters $S$. We trained the network using the algorithm mentioned in [16]. It can be
observed that by increasing the design parameter $S$, a much closer approximation to the ideal curve is obtained. The mean square error (MSE) obtained between the ideal and desired curve decreases from 0.00781 for $S = 1$ to 0.00034 for $S = 3$.

Table II compares the measured performance of S-AC based analog computing blocks presented in this work with similar designs reported in the literature. It can be observed that the implemented designs can function over a wide range of operating conditions (WI, MI, and SI) and at different power requirements with minimal area consumption when compared with similar technology node implementations. This is important to maintain the bias scalability of the designed circuit. This table shows the comparison of implemented ReLU activation with different variants of ReLU activations present in the literature. Table II also shows the comparison of this work with various four-quadrant full precision and approximate analog multipliers present in the literature. In addition, a comparison of log compressive DAC present in this work is also done with other DAC implementations present in literature whose inherent characteristic response is logarithmic in nature.

**VII. Conclusion**

In this work, we proposed S-AC based bias-scalable analog computing processor and near-memory S-AC core for machine learning (ML) tasks. We reported the basic building blocks (S-AC compressive memory DAC, S-AC ReLU, and S-AC multiply-accumulate) of the S-AC core and also showed the implementation of S-AC based compressive memory DAC, which also mimics the computation using Bfloat16 and IEEE 754 single-precision number systems. As a proof of concept, we demonstrated the implementation of a 3-layer S-AC neural network performing standard ML regression at different biasing conditions (signifying different operating speed and power consumption) and for different design parameter $S$ (signifying different computational accuracy).

In addition, S-AC based analog computing blocks were shown to remain invariant to biasing conditions and operating temperature. As a result, the S-AC based near-memory ML processor is well suited for high-speed training in the SI regime as well as for energy-efficient inference in WI regime, thereby allowing near-memory S-AC architectures to be used for both server and edge applications. It can be noted that a trade-off between speed and power can always be achieved.
by biasing in MI regime. At a system level, the overall performance (power and speed) of the S-AC processor can be adjusted by adjusting the hyper-parameter $C$ along with the input range, which in turn will bias the transistors in different operating regimes. In addition the design parameter $S$ allows the user to trade-off computational accuracy with the area and power [11]. We believe that this methodology can further be used to speed up other sub-variants of training algorithms such as physics-aware training [38] and hardware-algorithm co-design techniques [39] to provide a boost in overall systems efficiency. Our future works will include the demonstration of generic programmable architecture for deep neural networks.

**APPENDIX A**

**PROOF: S-AC IMPLEMENTATION OF ANALOG MULTIPLIER**

Consider the following equation, where $y$ is given by

\[
y = h(C + w + C + x) - h(C + w + C - x) + \ldots h(C - w + C - x) - h(C - w + C + x)\]  

(13)

The goal is to implement scalar multiplication between two variables $x$ and $w$. Here $x, w, y \in \mathbb{R}$, $h$ is a non-linear monotonic function and $C$ is a hyperparameter. If we write the Taylor expansion of $h(x)$ around $w$ and ignore the higher-order terms, we will get

\[
h(C + w + C + x) = h(C + w) + h(C + x) \Delta h(C + w) + \frac{C + x^2}{2} \Delta^2 h(C + w)\]  

(14)

\[
h(C + w + C - x) = h(C + w) + h(C - x) \Delta h(C + w) + \frac{C - x^2}{2} \Delta^2 h(C + w)\]  

(15)

\[
h(C - w + C - x) = h(C - w) + h(C - x) \Delta h(C - w) + \frac{C - x^2}{2} \Delta^2 h(C - w)\]  

(16)

\[
h(C - w - C + x) = h(C - w) + h(C + x) \Delta h(C - w) + \frac{C + x^2}{2} \Delta^2 h(C - w)\]  

(17)

Substituting (14) - (17) in (13) we get,

\[
y \approx [2x \Delta h(C + w) - 2x \Delta h(C - w)]\]  

(18)

\[
y \approx 2x \left[ \frac{dh(C + w)}{dw} - \frac{dh(C - w)}{dw} \right]\]  

(19)

\[
y \approx 2x (w^+ - w^-)\]  

(20)

\[
y \approx 2x \times w\]  

(21)

**ACKNOWLEDGMENT**

The authors would like to acknowledge the joint IISc-WashU MoU to facilitate the collaboration between the two institutions.

**REFERENCES**

[1] K. Freund. (Sep. 23, 2021). *IBM Research Says Analog AI will be 100X More Efficient. Yes, 100X* [Online]. https://www.forbes.com/sites/karlfreund/2021/09/23/ibm-research-says-analog-ai-will-be-100x-more-efficient-yes-100x/?sh=61b5ec23b129b

[2] M. Delmer. “Mythic multiples in a flash.” Linley Group Micro processor Rep., Aug. 2018. [Online]. Available: https://www.linleygroup.com/mp/issue.php?id=12023

[3] S. Chakraborty and G. Cauwenberghs, “Sub-microwatt analog VLSI trainable pattern classifier,” *IEEE J. Solid-State Circuits*, vol. 42, no. 5, pp. 1169–1179, May 2007.

[4] C. S. Thakur, R. Wang, T. J. Hamilton, J. Tapson, and A. V. Schaik, “A low power trainable neuromorphic integrated circuit that is tolerant to device mismatch,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 63, no. 2, pp. 211–221, Feb. 2016.

[5] S. Gupta, P. Kumar, T. Paul, A. van Schaik, A. Ghosh, and C. S. Thakur, “Low power, CMOS-MoS2 transistor based neuromorphic hybrid architecture for wake-up systems,” *Sci. Rep.*, vol. 9, no. 1, pp. 1–9, Oct. 2019.

[6] E. Vittoz and J. Fellrath, “CMOS analog integrated circuits based on weak inversion operations,” *IEEE J. Solid-State Circuits*, vol. SSC-12, no. 3, pp. 224–231, Jun. 1977.

[7] E. Seevinck and R. J. Wiegerink, “Generalized translinear circuit principle,” *IEEE J. Solid-State Circuits*, vol. 26, no. 8, pp. 1098–1102, Aug. 1991.

[8] M. Gu, “Theory, synthesis and implementation of current-mode CMOS piecewise-linear circuits using Margin Propagation,” Ph.D. dissertation, Dept. Elect. Eng., Michigan State Univ., 2012.

[9] M. Gu and S. Chakraborty, “Synthesis of bias-scalable CMOS analog computational circuits using margin propagation,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 2, pp. 243–254, Feb. 2012.

[10] C. Galup-Montoro, M. C. Schneider, A. I. A. Cunha, F. R. de Sousa, H. Klimach, and O. F. Siebel, “The advanced compact MOSFET (ACM) model for circuit analysis and design,” in *Proc. IEEE Custom Integr. Circuits Conf.*, Sep. 2007, pp. 519–522.

[11] S. Wang and P. Kanwar, “BFLOAT16: The secret to high performance on cloud TPUs,” Google Cloud, Aug. 2019. Accessed: May 2021. [Online]. Available: https://cloud.google.com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus

[12] A. R. Nair, P. K. Nath, S. Chakraborty, and C. S. Thakur, “Multiplication MP-kernel machine for energy-efficient edge devices,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 30, no. 11, pp. 1–14, Nov. 2022.

[13] T. P. Xiao, C. H. Bennett, B. Feinberg, S. Agarwal, and M. J. Marinella, “Analog architectures for neural network acceleration based on non-volatile memory,” *Appl. Phys. Rev.*, vol. 7, no. 3, Sep. 2020, Art. no. 031301.

[14] M. Gu, P. Kumar, T. Paul, A. van Schaik, A. Ghosh, and C. S. Thakur, “Margin decoding communication system,” *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 6, no. 1, pp. 1–10, Jan. 2022.

[15] F. J. Kub, K. K. Moon, I. A. Mack, and F. M. Long, “Programmable analog vector-matrix multipliers,” *IEEE J. Solid-State Circuits*, vol. 25, no. 1, pp. 207–214, Feb. 1990.

[16] C. R. Schlottmann and P. E. Hasler, “A highly dense, low power, programable analog vector-matrix multiplier: The fpa implementation,”* IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 1, no. 3, pp. 403–411, Sep. 2011.

[17] D. M. Binkley, “Tradeoffs and optimization in analog CMOS design,” in *Proc. 4th Int. Conf. Mixed Design Integr. Circuits Syst.*, Jun. 2007, pp. 47–60.

[18] Y.-H. Wu, W.-H. Lin, and S.-H. Huang, “Low-power hardware implementation for parametric rectified linear unit function,” in *Proc. IEEE Int. Conf. Consum. Electron.* (ICCE-Taiwan), Sep. 2020, pp. 1–2.

[19] P. Kumar, K. Zhu, X. Gao, S.-D. Wang, M. Lanza, and C. S. Thakur, “Hybrid architecture based on two-dimensional memristor crossbar array and CMOS integrated circuit for edge computing,” *NPJ 2D Mater. Appl.*, vol. 6, no. 1, pp. 1–10, Jan. 2022.

[20] D. M. Binkley, “Tradeoffs and optimization in analog CMOS design,” in *Proc. 14th Int. Conf. Mixed Design Integr. Circuits Syst.*, Jun. 2007, pp. 47–60.

[21] C. R. Schlottmann and P. E. Hasler, “A highly dense, low power, programable analog vector-matrix multiplier: The fpa implementation,” *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 1, no. 3, pp. 403–411, Sep. 2011.
[25] P. Priyanka, G. K. Nisarga, and S. Raghuram, “CMOS implementations of rectified linear activation function,” in VLSI Design and Test, S. Rajaram, N. B. Balamurugan, D. G. N. Rani, and V. Singh, Eds. Singapore: Springer, 2019, pp. 121–129.

[26] J. Zhu, Y. Huang, Z. Yang, X. Tang, and T. T. Ye, “Analog implementation of reconfigurable convolutional neural network kernels,” in Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS), Nov. 2019, pp. 265–268.

[27] C. Geng, Q. Sun, and S. Nakatake, “An analog CMOS implementation for multi-layer perceptron with ReLU activation,” in Proc. 9th Int. Conf. Modern Circuits Syst. Technol. (MOCAST), Sep. 2020, pp. 1–6.

[28] O. Krestinskaya, B. Choubey, and A. P. James, “Memristive GAN in analog,” Sci. Rep., vol. 10, no. 1, pp. 1–14, Apr. 2020.

[29] C. Popa, “Improved accuracy current-mode multiplier circuits with applications in analog signal processing,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 22, no. 2, pp. 443–447, Feb. 2014.

[30] R. J. D’Angelo and S. R. Sonkusale, “A time-mode translinear principle for nonlinear analog computation,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 62, no. 9, pp. 2187–2195, Sep. 2015.

[31] M. A. Al-Absi and I. A. As-Sabban, “A new highly accurate CMOS current-mode four-quadrant multiplier,” Arabian J. Sci. Eng., vol. 40, no. 2, pp. 551–558, Feb. 2015.

[32] I. Aloui, N. Hassen, and K. Besbes, “A CMOS current mode four quadrant analog multiplier free from mobility reduction,” AEU Int. J. Electron. Commun., vol. 82, pp. 119–126, Dec. 2017.

[33] J. Guilherme and J. Franca, “A logarithmic digital-analog converter for digital CMOS technology,” in Proc. Asia Pacific Conf. Circuits Syst. (APCCAS), 1994, pp. 490–493.

[34] S. Purighalla and B. Maundy, ”84-dB range logarithmic digital-to-analog converter in CMOS 0.18-µm technology,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 58, no. 5, pp. 279–283, May 2011.

[35] J. Zhang, Z. Wang, and N. Verma, “In-memory computation of a machine-learning classifier in a standard 6T SRAM array,” IEEE J. Solid-State Circuits, vol. 52, no. 4, pp. 915–924, Apr. 2017.

[36] L. Danial, N. Wainstein, S. Kraus, and S. Kvatinsky, “Breaking through the speed-power-accuracy tradeoff in ADCs using a memristive neuro-morphic architecture,” IEEE Trans. Emerg. Topics Comput. Intell., vol. 2, no. 5, pp. 396–409, Oct. 2018.

[37] M. G. Jomehei, S. Sheikhaei, E. H. Hafshejani, and S. Mirabbasi, “A low-power logarithmic CMOS digital-to-analog converter for neural signal recording,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 69, no. 1, pp. 15–19, Jan. 2021.

[38] L. G. Wright et al., “Deep physical neural networks trained with backpropagation,” Nature, vol. 588, no. 7869, pp. 549–555, 2022.

[39] J. Lee, S. Kang, J. Lee, D. Shin, D. Han, and H.-J. Yoo, “The hardware and algorithm co-design for energy-efficient DNN processor on edge/mobile devices,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 67, no. 10, pp. 3458–3470, Oct. 2020.

Pratik Kumar (Student Member, IEEE) received the M.Tech. degree from the Indian Institute of Technology (IIT), Dhanbad, India, in 2018. He is currently pursuing the Ph.D. degree with the Indian Institute of Science (IISc), Bengaluru, India. He is also associated with the NeuRomICS Laboratory, Department of Electronic Systems Engineering, and the Centre for Nano Science and Engineering, Indian Institute of Science. His current research interests include intersection of hardware-friendly machine learning algorithms, high-performance mixed-mode machine learning computational architectures, and multi-state emerging memory devices.

Ankita Nandi (Student Member, IEEE) received the B.Tech. degree in electronics and communication engineering from the National Institute of Technology Meghalaya, Shillong, India, in 2018, and the M.Tech. degree from the Indian Institute of Technology Gandhinagar, Gandhinagar, India, in 2020. She is currently pursuing the Ph.D. degree with the Department of Electronic Systems Engineering, Indian Institute of Science, Bengaluru, India. She was a recipient of the Fulbright Nehru Doctoral Research Fellowship from 2022 to 2023 and the Prime Minister’s Research Fellowship in 2021.

Shantanu Chakrabarty (Senior Member, IEEE) received the B.Tech. degree from the Indian Institute of Technology Delhi, India, in 1996, and the M.S. and Ph.D. degrees in electrical engineering from Johns Hopkins University, Baltimore, MD, USA, in 2002 and 2004, respectively. From 1996 to 1999, he was at Qualcomm Inc., San Diego, CA, USA. From 2004 to 2015, he was an Associate Professor at the Department of Electrical and Computer Engineering, Michigan State University (MSU), East Lansing, MI, USA. He is currently a Clifford W. Murphy Professor and the Vice-Dean for Research and Graduate Education with the McKelvey School of Engineering, Washington University in St. Louis, St. Louis, MO, USA. He was a recipient of the National Science Foundation’s CAREER Award, the University Teacher-Scholar Award from MSU, and the 2012 Technology of the Year Award from MSU Technologies. He has previously served as an Associate Editor for the IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS.

Chetan Singh Thakur (Senior Member, IEEE) received the Ph.D. degree in neuromorphic engineering from the MARCS Research Institute, Western Sydney University. He was a Post-Doctoral Researcher with Johns Hopkins University, Baltimore, MD, USA. He also worked as a Senior Integrated Circuit Design Engineer at Texas Instruments Singapore for few years. In 2017, he joined the Indian Institute of Science, Bangalore, as an Assistant Professor. His research interests include the computing principles of the brain and apply those to build novel intelligent VLSI systems. He received several awards, such as the Pratiksha Trust Young Investigator Award and the Inspire Faculty Award for brain-inspired computing from the Department of Science and Technology (DST).