2023 Design, Automation & Test in Europe Conference (DATE 2023)

Design of Large-Scale Stochastic Computing Adders and their Anomalous Behavior

Timothy Baker and John P. Hayes
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI, USA
{bakertim, jhayes}@umich.edu

***Abstract*— Stochastic computing (SC) uses streams of pseudo-random bits to perform low-cost and error-tolerant numerical processing for applications like neural networks and digital filtering. A key operation in these domains is the summation of many hundreds of bit-streams, but existing SC adders are inflexible and unpredictable. Basic mux adders have low area but poor accuracy while other adders like accumulative parallel counters (APCs) have good accuracy but high area. This work introduces parallel sampling adders (PSAs), a novel weighted adder family that offers a favorable area-accuracy trade-off and provides great flexibility to large-scale SC adder design. Our experiments show that PSAs can sometimes achieve the same high accuracy as APCs, but at half the area cost. We also examine the behavior of large-scale SC adders in depth and uncover some surprising results. First, APC accuracy is shown to be sensitive to input correlation despite the common belief that APCs are correlation insensitive. Then, we show that mux-based adders are sometimes more accurate than APCs, which contradicts most prior studies. Explanations for these anomalies are given and a decorrelation scheme is proposed to improve APC accuracy by 4x for a digital filtering application.**

Keywords—stochastic computing, weighted addition, large scale, design trade-offs, accuracy analysis, digital filtering

I. INTRODUCTION

Stochastic computing (SC) is a serial computing style that uses pseudorandom bit-streams called stochastic numbers (SNs) to encode and process data [1]. An SN $\mathbf{X} = x_1, x_2, \ldots, x_L$ is an L-bit sequence of pseudo-random bits that have the same probability of taking value 1: $P_X = \mathbb{P}(x_t = 1)$. The numerical value of $\mathbf{X}$ is derived from P_X based on the format used. In unipolar format $\mathbf{X}$'s value is $X = P_X$. In bipolar format, $\mathbf{X}$'s value is $X = 2P_X - 1$ which allows representation of negative numbers. The SN encoding enables the use of very small and power-efficient datapaths for computationally intensive applications like those in digital filtering [2][3][4] and neural networks [5][6][7][8][9][10][11].

The main attraction of SC is its relatively small circuits for multiplication and addition. Consider an AND gate with unipolar SN inputs $\mathbf{X}$ and $\mathbf{Y}$ and output $\mathbf{Z}$. The output SN's value $Z = P_Z = \mathbb{P}(z_t = 1)$ can be expressed as $\mathbb{P}\big((x_t = 1) \wedge (y_t = 1)\big)$ and simplified to $\mathbb{P}(x_t = 1)\mathbb{P}(y_t = 1) = XY$ provided $\mathbf{X}$ and $\mathbf{Y}$ are uncorrelated. Thus, $Z = XY$ implying that multiplication can be performed with just an AND gate once data are encoded into unipolar SNs. For bipolar SNs, an XNOR gate acts as an SN multiplier.

Like multiplication, adding SNs can also be low in cost [1]. For example, the two-way mux in Fig. 1b performs scaled addition $Z = 0.5(X + Y)$ on its data inputs $\mathbf{X}$ and $\mathbf{Y}$ by using a control SN $\mathbf{S}$ with probability $P_S = 0.5$. In each clock cycle, $\mathbf{S} = s_1 s_2 \ldots s_L$ determines whether $\mathbf{X}$ is sampled (i.e., when $s_t = 0$) or $\mathbf{Y}$ is sampled (i.e., $s_t = 1$). Over the long run, half of $\mathbf{Z}$'s bits will be sampled from $\mathbf{X}$ and half from $\mathbf{Y}$ implying that $\mathbf{Z}$'s value is the average of $\mathbf{X}$'s and $\mathbf{Y}$'s values.

This research was supported by the U.S. National Science Foundation under Grant CCF-2006704.

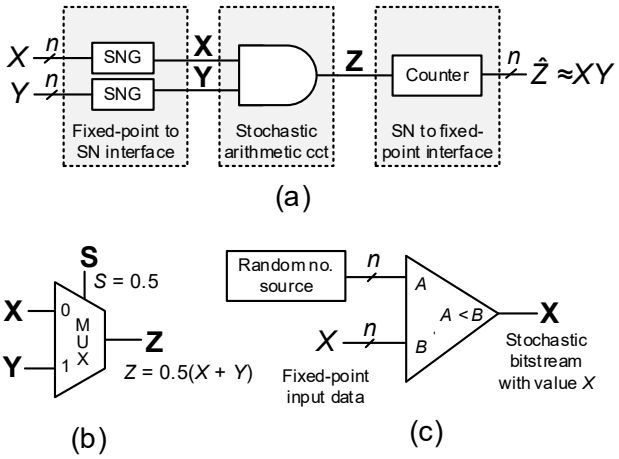

Fig. 1. (a) SC arithmetic circuit embedded in traditional computing system. (b) Multiplexer (mux) based SN adder; (c) SN generator (SNG).

A tree of two-way muxes can be used to implement a large adder with many inputs. However, such mux tree adders, which exist in many variants [3][4], usually have poor accuracy when used to sum hundreds of inputs [2][5]. Much research has been devoted to implementing large-scale stochastic addition, and many alternatives to mux adders have been proposed. A common alternative adder is the accumulative parallel counter (APC) [12] which counts in parallel all the 1s appearing in its input SNs in each clock cycle. APCs are usually more accurate than mux adders, but also have much higher area cost due to their counting approach.

Besides APCs, other novel SC adder types have been proposed [2][7][8][9]. A cursory analysis reveals that existing SC adders tend to belong to one of two extremes; either the adder is accurate but large, or else it is small but inaccurate. To increase the flexibility of SC adder design, this work introduces parallel sampling adders (PSAs). The PSA framework greatly improves SC adder design flexibility by offering a smooth accuracy-area trade-off. The trade-off is very favorable in some cases where PSAs achieve the same accuracy as APCs while using half the area. This work also examines the behavior of large-scale SC adders in depth and finds some surprising results. The main contributions are:

1. Introduction of the parallel sampling adder which adds great flexibility to the design of large-scale SC adders.
2. Demonstration that correlation impacts APC accuracy despite the common belief that APCs are correlation insensitive. A low-cost decorrelation scheme is presented for improving APC accuracy.

978-3-9819263-7-8/DATE23/© 2023 EDAA

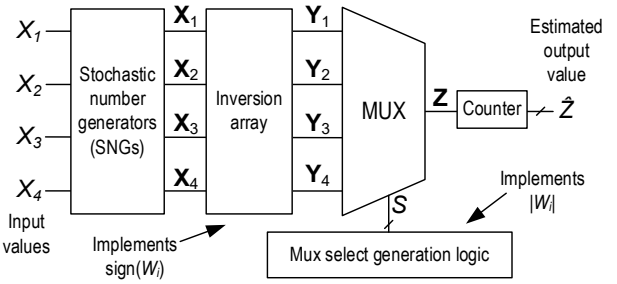

Fig. 2. Four-input bipolar mux adder that implements (1) with scale factor $\alpha = \sum_{i=1}^{4}|W_i|$. There are various ways to configure the mux select generation logic [3][4]. For example, the CeMux design [2] uses a plain binary counter for the mux select generation logic**.**

3. A comprehensive comparison of large-scale adder designs leading to some surprising results like mux adders sometimes have better accuracy than APCs.

The rest of the paper is organized as follows: Sec. II reviews relevant SC concepts while Sec. III then introduces the novel PSA design. Sec. IV details case studies that demonstrate the usefulness of PSAs and reveal surprising behavior of large-scale SC adders. Sec. V draws conclusions.

II. Stochatsic Computing Adders

A. Stochastic Computing Basics

To take advantage of low-cost SC multiplication and addition circuits, data must first be converted into SN bit-streams. The structure of an SC system intended to be embedded in a larger, conventional computing system is sketched in Fig. 1a. Here, SN generators (SNGs) are first used to encode data into SNs. Specifically, SNs **X** and **Y** are generated with corresponding values X and Y. Then **X** and **Y** are multiplied using an AND gate to produce $\mathbf{Z} = z_1 z_2 \ldots z_L$ with numeric value $Z = XY$. Finally, a counter is used to estimate **Z**'s value as $\hat{Z} = \frac{1}{L}\sum_{t=1}^{L} z_t$ over L clock cycles.

It is important to note the distinction between the various values associated with an SN such as **Z**. There is the SN's expected value $Z = \mathbb{E}[\hat{Z}]$ determined by the circuit design and randomness sources. There is also **Z**'s estimated value $\hat{Z}$ found by counting the 1s in **Z**, and **Z**'s target value Z^* determined by the application. It is often the case that $Z = Z^*$, but $\hat{Z} \neq Z^*$ due to the inherent randomness of SC. The difference between $\hat{Z}$ and Z^* is the SN's error $\epsilon_Z = \hat{Z} - Z^*$.

One design challenge of SC is that a stochastic circuit's output $\hat{Z}$ is a probabilistic approximation of the target output value Z^*. The accuracy of $\hat{Z}$ depends on the bit-stream length L, the circuit design, and possibly other factors [14]. Generally, accuracy improves as L increases because random fluctuations average out over many clock cycles. However, using very large L means slow operation and high energy consumption. Thus, a major focus of stochastic circuit design is to improve accuracy for a given SN length [13][14].

Another design challenge of SC is that generating SNs is very costly compared to performing SN arithmetic. As suggested in Fig. 1c, SNGs are large and typically consist of a comparator alongside a pseudo-random number source like a linear feedback shift register [1] or Sobol sequence generator [13]. To amortize SNG overhead, SC is best used to implement expensive operations like the large-scale weighted summations found in digital filters or neural networks.

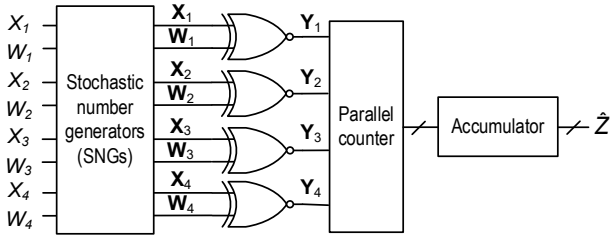

Fig. 3. APC-based bipolar weighted adder. The XNOR gates multiply the input SNs by the weight SNs. The parallel counter accumulates the products.The accumulator stores the sum over the entire SN length.

B. Stochastic Computing Adder Design

SC weighted adders implement

$$Z^* = \frac{1}{\alpha}\sum_{i=1}^{M} W_i X_i \qquad (1)$$

where X_i are the input values with corresponding weight W_i and α is a scale factor which depends on the adder design.

Multiplexers are the traditional weighted SC adder type. Fig. 2 illustrates a bipolar mux adder with $M = 4$ inputs. Input values X_1 to X_4 are first encoded into bipolar SNs $\mathbf{X}_1$ to $\mathbf{X}_4$. The inverter array with output $\mathbf{Y}_1$ to $\mathbf{Y}_4$ then implements the sign of the input weights. If $W_i < 0$, then $\mathbf{X}_i$ is inverted and $\mathbf{Y}_i$ has value $Y_i = -X_i$, otherwise $\mathbf{X}_i$ is left unchanged and $\mathbf{Y}_i$ has value $Y_i = X_i$. Next, the mux samples one of its inputs. The probability that $\mathbf{Y}_i$ is sampled is made equal to its normalized weight $|W_i|/\sum_{i=1}^{4}|W_i|$ and this sampling value effectively implements the magnitude of the weights. The mux output **Z**'s value is a weighted sum (1) with $M = 4$ and $\alpha = \sum_{i=1}^{4}|W_i|$.

Conventional mux adders require very long bitstreams to achieve practical accuracy thresholds [2][3]. For instance, when implementing a 267-input weighted summation, basic mux adders required 2^{2n+1}bit SNs to match the accuracy of a conventional n-bit binary design [3]. Thus, to match 8-bit binary adder's performance, SNs exceeding 100,000 bits in length are needed and lead to high latency and energy consumption. Recently, however, a mux adder named CeMux [2] was introduced that has about 30% lower area and about 9x lower error than basic mux designs. CeMux achieves higher accuracy and lower cost by ensuring all mux inputs are maximally correlated and by using a deterministic sampling process that does not bias the output [2].

Another common SC weighted adder type employs APCs. Fig. 3 illustrates a 4-input bipolar weighted APC. Here, the input values X_1 to X_4 and the corresponding weights W_i are converted into bipolar SNs that are multiplied using XNOR gates. The product SNs $\mathbf{Y}_1$ to $\mathbf{Y}_4$ with value $Y_i = W_i X_i$ are then summed using an accumulative parallel counter. The APC output is $\hat{Z}$ with value $\sum_{i=1}^{4} Y_i = \sum_{i=1}^{4} W_i X_i$. Overall, APC weighted adders implement (1) with no scale factor ($\alpha = 1$). APCs (sometimes called population counters) exhaustively count all 1s from the product SNs in each clock cycle, and thus have high accuracy as well as high cost. APC and mux adders are the most common SC adders; other adder designs are reviewed in [5] and discussed briefly in Secs. III and IV.

Normally, every input and weight SN requires its own SNG (Fig. 1c) which can be costly in terms of area and power. However, cost can be greatly reduced by sharing a single RNS across several SNGs as in Fig. 4a. Such RNS sharing leads to correlation amongst the generated SNs, and careful design is

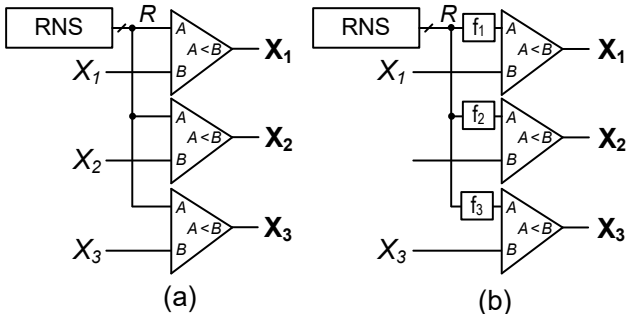

Fig. 4. Sharing a single RNS amongst multiple SNGs. (a) direct sharing; (b) sharing where each SNG applies a randomly chosen network of inverters to the shared RNS value. The inverter networks are represented by the f_i boxes.

required to avoid correlation-induced error. For mux adders, a single RNS is shared amongst all input SNGs which has been shown to improve accuracy [2]. For weighted APCs, one RNS is shared amongst all input SNGs and a second RNS is shared amongst all weight SNGs. Generating inputs and weights with separate RNSs ensures input and weight SNs are independent which is required for accurate XNOR multiplication.

III. Parallel Sampling Adders

Parallel sampling adders (PSAs) combine mux adders and APCs into a single flexible design. In a PSA, input SNs are arranged into disjoint groups of size G, each of which is applied to a separate mux adder. The outputs of all muxes are then accumulated by an APC. For example, Fig. 5a shows an 8-input PSA with two groups of size $G = 4$. Both groups are summed separately by muxes with outputs $\mathbf{Y}_1$ and $\mathbf{Y}_2$ that have value $Y_1 = 0.25\ \sum_{i=1}^{4} X_i$ and $Y_2 = 0.25 \sum_{i=5}^{8} X_i$, respectively. Left bit-shifts S are then used to remove the 0.25 scale factors from the mux output values before $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are accumulated by a small APC whose output value is $Z = \sum_{i=1}^{8} X_i$.

A PSA is parameterized by its group size G which is the maximum input size of any mux adder used in the PSA. For example, the PSA in Fig. 5a has $G = 4$ while the PSA in Fig. 5b has $G = 2$. Group size represents the degree of approximation of the PSA – higher G means a more aggressive approximation and typically lower area and accuracy. For an M-input PSA, G takes values between 1 and M and is restricted to be a power-of-two so that scale factors introduced by the mux adders can always be removed by bit-shifts. All muxes in a PSA are implemented as CeMux designs because it is the smallest and most accurate mux adder [2].

A. Weighted PSAs

The basic PSA design in Fig. 5 is now extended to implement general weighted addition (1) with $\alpha = 1$. Fig. 6 displays a bipolar weighted PSA with $M = 7$ inputs and group size $G = 4$. First, inputs and weight SNs are multiplied using XNOR gates to produce 7 product SNs $\{\mathbf{P}_i\}_{i=1}^{7}$ with value $P_i = W_i X_i$. The product SNs are then grouped into sets of size $G = 4$. In this case, one group is constructed: $\{\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4\}$. Three SNs remain that cannot form another four-element group, so G is cut in half and one group of size 2 is constructed: $\{\mathbf{P}_5, \mathbf{P}_6\}$. As one SN remains, the group size is again cut in half to size one and a final group $\{\mathbf{P}_7\}$ is formed. Each group is input to a separate mux and all mux outputs are summed with an APC to produce the final output $\hat{Z}$.

One viewpoint on PSA design is that it uses mux adders to approximate and reduce the size of a pure APC design. PSAs may therefore seem superficially similar to approximate APC designs like AxPC [15] and SUC adders [7], but they actually differ in fundamental ways. Unlike AxPC and SUC adders, PSAs can be used with bipolar SNs. PSA weights also do not need to be partitioned into disjoint subsets that sum to 1 as in SUC adders. Lastly, PSA weights are programmable, whereas changing weights in an SUC adder requires redesigning hardwire interconnections.

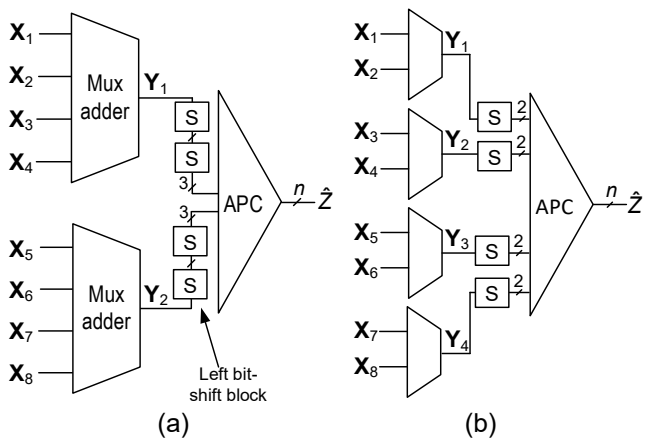

Fig. 5. Basic PSAs that implement $Z = \sum_{i=1}^{8} X_i$. S denotes a 1-bit shift left. (a) PSA with group size $G = 4$ (b) PSA with group size $G = 2$.

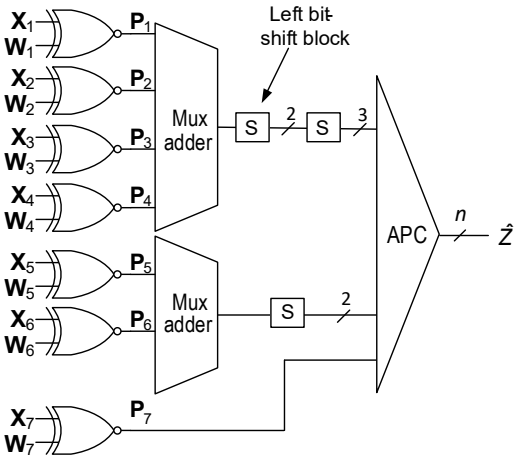

Fig. 6. Weighted PSA with group size $G = 4$. An array of XNOR gates is first used to multiply the bipolar inputs and weights. Product SNs are then accumulated by mux adders of various sizes. All scale factors introduced by the mux adders are removed with left bit-shifts S before the mux outputs are accumilated by a small APC whose exepected output is $Z = \sum_{i=1}^{7} W_i X_i$.

B. Area-Accuracy Trade-off

A PSA's area and accuracy depend on the PSA's group size G. Larger G implies a higher level of approximation and thus lower area and accuracy. To determine how area changes with G we used Synopsys Design Compiler (DC) with the Nangate 45nm library [16] to synthesize PSAs with various sampling group sizes, input sizes, and SN lengths.

Fig. 7a shows the area of a PSA configured to perform weighted addition on 512 unipolar SNs of length 128. Here, area varies roughly linearly with group size G. When $G = 1$, the PSA is equivalent to a pure APC adder and its area is about 70% higher than the smallest PSA design with $G = 512$. These results make intuitive sense: as sampling group size increases, the size of the APC in the PSA shrinks proportionally thus decreasing area cost in a consistent manner. Other SN lengths and input sizes were tested and yielded similar normalized area results.

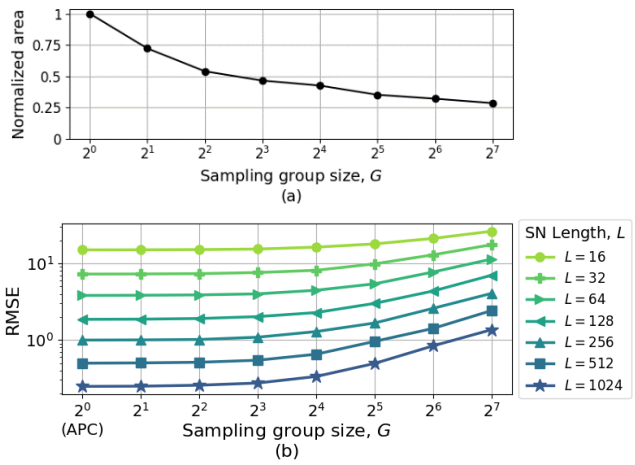

Fig. 7. PSA area and accuracy trade-offs: (a) Unipolar PSA synthesized area as a function of group size. Area is normalized by dividing by a weighted APC's area; (b) Unipolar PSA error vs. group size.

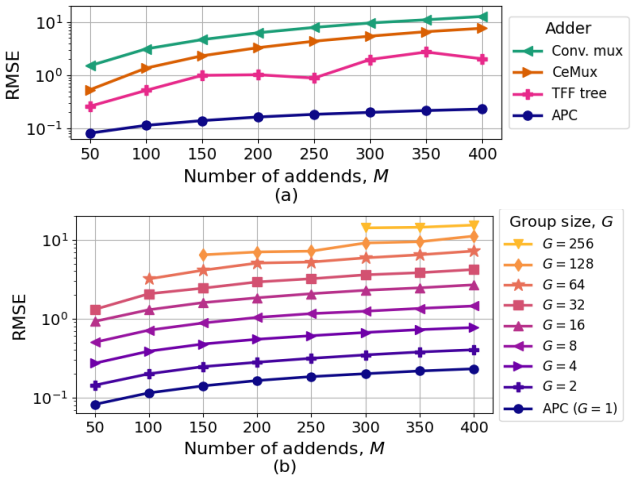

Fig. 8. Error vs. number of addends for (a) various bipolar SC adders; (b) bipolar PSA adders. Both plots have the same y-axis limits.

The impact of group size G on accuracy was investigated by simulating a unipolar 512-input PSA with SN length L. For each pair of values G and L, the PSA is simulated with uniformly random input and weight values $X_i, W_i \in [0,1]$. Root mean square error (RMSE) is estimated as

$$\text{RMSE} = \sqrt{\frac{1}{\text{R}}\sum_{r=1}^{\text{R}}\left(\frac{\hat{Z}_r - Z_r^*}{\alpha}\right)^2} \qquad (2)$$

where $\alpha = 1$ is the PSA's scale factor, $\hat{Z}_r$ is the PSA's output during a simulation run r, Z_r^* is the PSA's target value found using (1) during run r, and $R = 10{,}000$ simulations are used to yield statistically significant results. Note that RMSE is normalized by the adder's scale factor α to facilitate fair comparisons later in Sec. IV when adders with different α, and thus different sensitivities to error magnitude, are compared.

Fig. 7b plots the simulation results. Interestingly, all PSAs with $G \leq 16$ have nearly the same RMSE as an APC, which is equivalent to a PSA with $G = 1$. This finding is significant because a PSA with $G = 16$ requires about 50% less area than an APC, as shown in Fig. 7a. For $G \geq 32$, the RMSE increases roughly proportionally with G, so area can be traded for accuracy by adjusting G. The SN length L only affects the magnitude of RMSE but does not affect the shape of the RMSE vs. group size curve. Thus, the accuracy trade-off is very consistent across different SN lengths.

IV. Case Studies

Here, PSAs are compared against other high-performing bipolar SC weighted adders in terms of area and accuracy for large-scale weighted addition. The designs considered include conventional mux trees (conv. mux) [3], CeMux [2], APC [12], and a tree of T-flip-flop (TFF) adders [8]. To maximize accuracy and minimize area, all the designs use a shared Sobol RNS [13] for input SN generation. The APC, PSA and TFF tree designs each use a second shared Sobol RNS to generate their weight SNs, as is typical.

A. Baseline

First, each adder's accuracy is assessed for general large-scale bipolar weighted addition by testing each design with random input and weight values. Each design uses 256-bit SNs to perform weighted addition on M inputs where M is varied. For each value of M, the design is simulated $R = 10{,}000$ times with uniformly random input values and weights, $X_i, W_i \in [-1,1]$. The RMSE estimated using (2) is plotted as a function of input size in Fig. 8.

Fig. 8a demonstrates that the most accurate non-PSA design is the APC followed by the TFF tree and then CeMux. The conventional mux adder has the highest RMSE and thus has the worst accuracy. Fig. 8b illustrates how the RMSE varies for PSAs with various group sizes. As expected, as group size is increased, the PSA's error in terms of RMSE smoothly rises for all input sizes. Thus, the PSAs accuracy and area can be finely tuned by adjusting G.

B. Neural Network Case Study

Large-scale SC adders have been used to design low-power neural network (NN) ASICs [5] which rely heavily on the weighted summation operation (2). One such design [10] uses SC to implement the first layer of a binarized NN (BNN). BNNs are a class of NNs where all weights and activations are binarized to take values ± 1. This extreme quantization facilitates the design of low-cost NNs while maintaining high classification accuracy on image classification benchmarks.

When used for image processing, BNNs often do not binarize the raw (grayscale) pixel inputs because too much information would be lost. Instead of employing traditional fixed-point computing, implementing the first BNN layer with SC and popcount circuits (i.e., APCs) leads to a 62% reduction in overall area for the two-layer network in [10]. Here we investigate the extent to which PSAs can be used to further improve hardware efficiency while maintaining classification accuracy.

As in [10], we apply a binarized multi-layer perceptron (MLP) with two 1,024-neuron hidden layers to the Fashion-MNIST benchmark [17]. Fashion-MNIST consists of 28x28 grayscale images of fashion items that fall into one of ten classes like "coat" or "sneaker" – eexamples are shown in Fig. 9a. The hybrid SC-BNN network employs PSA adders in the first layer and the iterative training procedure from [10] was used to train the network.

After training, the network is evaluated on the 10,000 test images from the Fashion-MNIST dataset. The PSA's group size G is varied while the RMSE (2) of the first layer implemented with SC and the network's overall classification

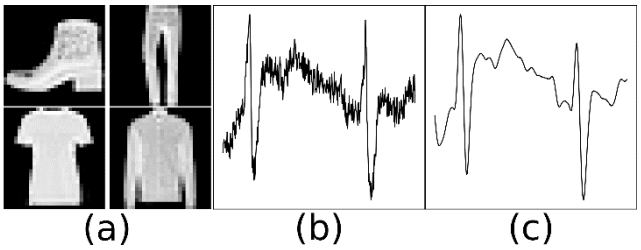

Fig. 9. Case study data: (a) example images from Fashion-MNIST classification benchmark; (b) noisy ECG signal; (c) filtered ECG signal.

accuracy are measured. The SN length was set to 16 bits which is the shortest length that gave the maximum classification accuracy of about 88%, which matches the performance of a non-SC version of the network. The area of a first-layer neuron is measured for each value of G using Synopsys DC with the Nangate 45nm library [16] and area is normalized by dividing by the area of a neuron that employs an APC adder. Since SNGs can be shared across many neurons, normalized neuron area is given both with and without SNGs. All results are presented in Table I.

Table I shows that the first layer's RMSE increases roughly in proportion with group size G. For $G = 2$ and $G = 4$, the classification accuracy varies slightly compared to using an APC, but area is much lower. Using a PSA with $G = 4$ in place of an APC reduces the neuron area by half while only reducing classification accuracy by 0.26%. For larger group sizes like $G = 8$ and $G = 16$, the classification accuracy drops by 1.6% and 3% in exchange for further area savings. Group sizes beyond $G = 16$ lead to poor results since the SN length is only 16 bits.

A hybrid SC-BNN network that employs pure CeMux adders in the first layer is also evaluated. The network's classification accuracy is only 25% because 16-bit SN length is short relative to the number of inputs 784. In contrast, the PSA avoids inaccuracy problems by employing several small mux adders in parallel while saving area compared to an APC.

C. ECG Filtering

The World Health Organization estimates that 17.9 million people died of cardiovascular disease in 2019, representing about 32% of global deaths [18]. Low-power continuous heart rate monitoring can help to better address the risk of cardiovascular disease. A key preprocessing step is the denoising of the electrocardiogram (ECG) signal as in Fig. 9b and 9c [19]. Denoising can be performed by finite impulse response (FIR) filters. An M-tap FIR filter implements

$$y_t = \sum_{k=0}^{M-1} h_k x_{t-k} \tag{3}$$

where $\{x_t\}$ is the noisy input signal, $\{h_k\}$ are the M filter coefficients and $\{y_t\}$ is the filtered output signal. Generally, FIR filters with more taps (higher M) perform better filtering at the cost of more computational resources. Due to their large computational needs, SC has been proposed as a low-cost solution to FIR filtering [2][3][4].

Here, each large-scale bipolar SC adder is used to implement an FIR filter that denoises ECG signals from Physiobank's MIT arrhythmia database [20]. Random noise is added to the benchmark signals to simulate three major noise types: device noise, electrosurgical noise, and noise from muscle contractions [21]. Fig. 10 plots the RMSE of each SC adder against the number of filter taps M when 256-bit SNs are used. Compared to the baseline test of Fig. 8, the APC's RMSE is 3.2x to 4.4x higher depending on M. Meanwhile, CeMux's error has decreased by 35x to 144x compared to the baseline test. Similar results were found, but not explored in depth in [2]. Overall, the relative ranking of adder accuracy for filtering ECG signals differs significantly from the baseline test with random data. The next section ventures to explain this strange result.

TABLE I. PSA-BASED NEURAL NETWORK PERFORMANCE

Group Size, G	First Layer RMSE	Classification accuracy (%)	Normalized area w/ SNGs	Normalized area w/o SNGs
1 (APC)	1.83	87.26	1	1
2	2.23	87.86	0.73	0.74
4	3.93	87.00	0.56	0.50
8	7.43	85.66	0.47	0.35
16	13.26	84.22	0.45	0.24

D. Anomalous Behavior

An important, but sometimes overlooked influence on circuit accuracy is the application's input value distribution [14]. For instance, [14] found that multiplication error in an SC NN differed by as much as 2.3x depending on which benchmark dataset was used. In the case of ECG filtering, the input SNs have similar values because they are derived from a continuous ECG signal like that of Fig. 9b and this similarity of input values leads to higher mux accuracy [2]. Thus, CeMux's accuracy for ECG filtering is substantially (35x to 114x) higher than a baseline test with random data suggests.

On the other hand, the APC's surprisingly poor accuracy on ECG filtering is due to a combination of input value distribution and bitstream correlation from RNS sharing. Consider the APC of Fig. 3. Since the APC counts all 1s of the product SNs $\mathbf{Y}_1$ to $\mathbf{Y}_4$, the APC's error $\epsilon_Z = \hat{Z} - Z^*$ can be expressed as the sum of multiplication errors $\epsilon_{Y_i} = \hat{Y}_i - Y_i^*$ where $Y_i^* = W_i X_i$. When no RNS sharing is used, the product SNs are independent and about half the multiplication errors ϵ_{Y_i} are positive and half are negative during any given simulation run. These errors partially cancel out resulting in a low overall summation error ϵ_Z. In contrast, when RNS sharing is employed, the product SNs $\mathbf{Y}_i$ are correlated. In this case, the multiplication errors have similar magnitude as in the no-RNS-sharing case, but the errors are correlated and tend to have the same sign. Consequently, the multiplication errors cancel less often than when there is no RNS sharing, and the overall summation error is much higher.

The APC's accuracy can be improved by decorrelating its inputs through not sharing RNSs, but this is prohibitively costly. Instead, an RNS sharing scheme like that of Fig. 4b is employed. Here, SNGs share an RNS, but each SNG inverts a different subset of the RNS bits. The inversions are chosen randomly but are fixed once chosen and partially decorrelate the generated SNs [11]. Random permutations could also be

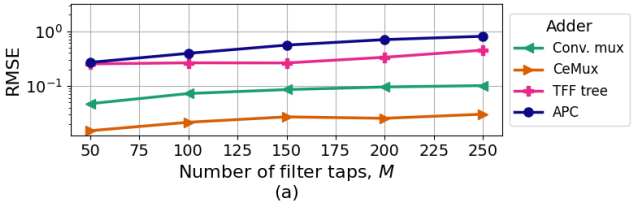

Fig. 10. ECG filtering error vs. input size for various bipolar SC adders.

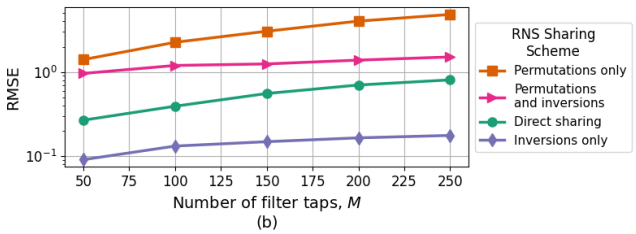

Fig. 11. ECG filtering error for APCs with various RNS-sharing schemes.

applied to a shared RNS state for decorrelation purposes [4][11][22]. However, when random permutations successfully decorrelate multiplication errors as desired, they simultaneously disrupt the special low-discrepancy properties of the Sobol RNS which tends to increase the magnitude of the multiplication errors. The overall effect is that random permutations increase summation error as the following data suggests.

Fig. 11 plots the weighted APCs accuracy for ECG filtering for various RNS sharing schemes. As before, all schemes share one Sobol RNS amongst the input SNGs and share one Sobol RNS amongst the weight SNGs, but each scheme differs in whether random permutations or inversions are applied to the shared RNS state. Compared to direct sharing, sharing with inversions greatly decreases RMSE by 3.4x to 4.5x depending on input size whereas sharing with permutations or with permutations and inversions increases RMSE as suggested earlier. With inversion-based decorrelation, the APC's accuracy now matches its performance on the baseline, but still has higher RMSE than CeMux for ECG filtering.

Although inversion-based decorrelation significantly improves the APC's accuracy in the ECG filtering case, it does not affect APC accuracy in either the earlier random baseline test or the Fashion-MNIST case study. In those cases, the multiplication errors are uncorrelated even when direct RNS sharing is used, and so the decorrelation scheme does not influence accuracy. The sensitivity of APCs to input correlation therefore depends on the application's input value distribution. Since the inversion-based decorrelation either decreases RMSE or leaves, it unchanged compared to direct sharing, the technique appears generally useful given that inverters are low-cost and can be absorbed into the SNGs' comparators.

Overall, CeMux has the lowest error for ECG filtering, which corroborates the results of [2]. CeMux's superior accuracy is surprising since mux-based adders usually do not outperform APCs. One drawback of CeMux, however, is that weights are hardwired into the design. In the case that programmable weights are desired, APCs or PSAs can be used to achieve an appropriate balance between area and accuracy.

In summary, the ECG filtering case study illustrates two important and surprising behaviors of large-scale adders. First, RNS sharing can impact the accuracy of APCs which contradicts common belief [5][6]. Our proposed decorrelation scheme with random inversions can decrease this correlation and restore APC accuracy. Second, mux adders can sometimes outperform APCs as first demonstrated in [2] and investigated further here. CeMux's accuracy improved by 35x to 144x compared to the random baseline study while APC's accuracy, after decorrelation remained roughly the same.

V. Conclusion

SC offers the promise of low-cost large-scale weighted adders. Here, we introduced the PSA adder which adds great flexibility to SC adder design and leads to a 50% reduction in SC neuron area in a hybrid SC-BNN model. Further investigation of SC adders highlighted the importance of input value distribution and the limitations of random-input-based baseline studies. For some applications, mux adders can outperform APCs. Furthermore, APCs are sometimes correlation sensitive, but can be successfully decorrelated by adding randomly chosen networks of inverters to the SNGs.

References

[1] B.R.Gaines, "Stochastic computing systems," *Advances in Information Systems Science*, Springer New York, 37-172, 1969.

[2] T.J. Baker and J.P. Hayes, "CeMux: Maximizing the accuracy of stochastic mux adders and an application to filter design," *ACM Trans. Design Auto. Elec. Sys, 27*, **3**, 1-26, 2022.

[3] R. Wang, et al., "Design, evaluation and fault-tolerance analysis of stochastic FIR filters," *Microelectronics Reliability*, **57**, 111-127, 2016.

[4] H. Ichihara, et al., "Compact and accurate digital filters based on stochastic computing," *IEEE TETC,* **7**, 1, 31-43, 2019.

[5] Y. Liu, et al., "A survey of stochastic computing neural networks for machine learning applications," *IEEE Trans. Neural Networks and Learning Systems*, **32**, 7, 2809-2824, 2020.

[6] S.R. Faraji, et al., "Energy-efficient convolutional neural networks with deterministic bit-stream processing," *Proc. DATE*, 1757-1762, 2019.

[7] B. Li, M.H. Najafi, and D.J. Lilja, "Low-cost stochastic hybrid multiplier for quantized neural networks". *J. Emerg. Technol. Comput. Syst*. **15**, 2, 1-19, 2019.

[8] V.T. Lee, et al., "Energy-efficient hybrid stochastic-binary neural networks for near-sensor computing," *Proc. DATE,* 13-18, 2017.

[9] A. Zhakatayev, et al., "Sign-magnitude SC: Getting 10X accuracy for free in stochastic computing for deep neural networks," *Proc. DAC,* 1-6*,* 2018.

[10] T. Hirtzlin, et al., "Stochastic computing for hardware implementation of binarized neural networks," *IEEE Access*, **7**, 76394-76403, 2019.

[11] Y. Xie, et al., "Fully-parallel area-efficient deep neural network design using stochastic computing," *IEEE TCAS II*, **64**, 1382-1386, 2017.

[12] B. Parhami and C-H. Yeh, "Accumulative parallel counter," *Proc. Asilomar Conf. on Signals, Systems and Computers*, **2**, 966-970, 1995.

[13] S. Liu and J. Han, "Energy efficient stochastic computing with Sobol sequences," *Proc. DATE*, 650–653, 2017.

[14] T.J. Baker and J.P. Hayes, "Bayesian accuracy analysis of stochastic circuits," *Proc. ICCAD*, 1-9, 2020.

[15] K. Kim, J. Lee and K. Choi, "Approximate de-randomizer for stochastic circuits," *Proc. ISOCC,* 123-124, 2015.

[16] J.E. Stine, et al., "FreePDK: An open-source variation-aware design kit," *Proc. IEEE MSE*, 173-174, 2007.

[17] H. Xiao, et al., "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms." arXiv:1708.07747, 2017.

[18] World Health Organization, "Cardiovascular Diseases," www.who.int/health-topics/cardiovascular-diseases/ (accessed Sept. 2022).

[19] M.A. Serhani, et al., "ECG Monitoring Systems: Review, Architecture, Processes, and Key Challenges," *Sensors*, **20**, 2020.

[20] G.B. Moody and R.G. Mark, "The impact of the MIT-BIH Arrhythmia Database" *IEEE Eng in Med and Biol*, **20**, 45-50, 2001.

[21] G.M. Friesen, et al., "A comparison of the noise sensitivity of nine QRS detection algorithms," *IEEE Trans. Bio. Eng.*, **37**, 85-98, 1990.

[22] S.A. Salehi, "Low-cost stochastic number generators for stochastic computing," *IEEE Trans. VLSI*, **28**, 992-1001, 2020.