In-Stream Correlation-Based Division and Bit-Inserting Square Root in Stochastic Computing

Di Wu, Ruokai Yin, and Joshua San Miguel
University of Wisconsin–Madison,
Madison, WI 53706 USA

Editor's notes:
This article presents improved stochastic computing primitives for division and square root operations. Both are nonlinear functions that cannot be reduced to additions and multiplications. The authors make use of the very correlations that are usually considered undesirable in stochastic computing; their controlled injection into computations leads to good compromises between convergence time and area requirements.
—Weikang Qian, Shanghai Jiao Tong University

GAINES INITIALLY PROPOSED stochastic computing (SC) [1] as a low power solution for applications requiring massive but often redundant inputs such as machine learning and pattern recognition. Due to its extremely simple computing logic, like an AND gate for multiplication, as shown in Figure 1b, SC has regained research interest in error-correcting codes and computer vision in the past decades [2]. More recently, with the evolution of artificial intelligence, SC further finds popularity in varying deep learning models containing heavy matrix operations [2]. Bernoulli sequences, containing uniformly distributed 0's and 1's, are usually used as SC data, namely bit stream (BS). Its value is exclusively determined by the probability of 1's in the BS, with precision relying on the BS length. There are two fundamental SC data representations, namely unipolar and bipolar. Unipolar data have the unsigned value, V_{Uni} of range [0,1], equal to the probability of 1's in the BS, P_{Uni}, while bipolar data have the signed value $V_{\text{Bi}} = 2 \times P_{\text{Bi}} - 1$ of range [–1,1], with P_{Bi} referring to the probability of 1's. Figure 1a shows unipolar SC data representation. A and B have the same value of 0.5 despite different element orders, while C has a larger value due to 4 more 1's. Thanks to this bit-serial representation, SC is a promising computing paradigm for emerging deep learning models that are computationally intensive but require extremely low area and power.

Though simple in hardware, SC introduces extra computation latency and fluctuation in accuracy. For example, with length-N BSs, unipolar SC multiplication (Figure 1b) requires N cycles, significantly higher than that for binary multiplication. Furthermore, with varying input BSs, the output varies with the stochastic cross correlation [3] between inputs (i.e., how aligned the 1's and 0's are). Figure 1c shows how a high positive correlation affects the output of the AND gate, and leads to a MIN operation. To achieve high accuracy, most existing works focus on a vital but expensive component of SC circuits, BS generation, which generates the Bernoulli sequence [1] by producing a sequence of random numbers (RNs) and comparing them each to an input value. Conventional techniques aim to construct zero-correlation BSs using high-quality RN generators (RNGs), like bit scrambling [4] and low-discrepancy sequences [5]. Only recently

Digital Object Identifier 10.1109/MDAT.2021.3050716
Date of publication: 11 January 2021; date of current version: 6 December 2021.

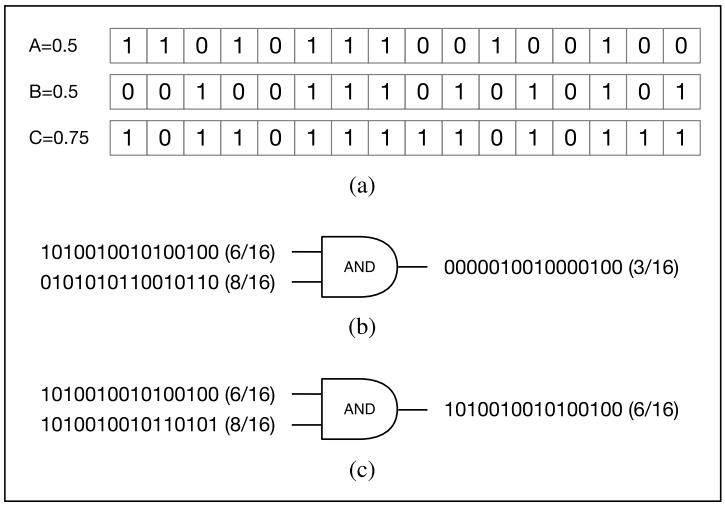

Figure 1 Unipolar SC paradigm. (a) Unipolar representation. (b) Unipolar multiplication (zero correlation). (c) Unipolar minimum (high positive correlation).

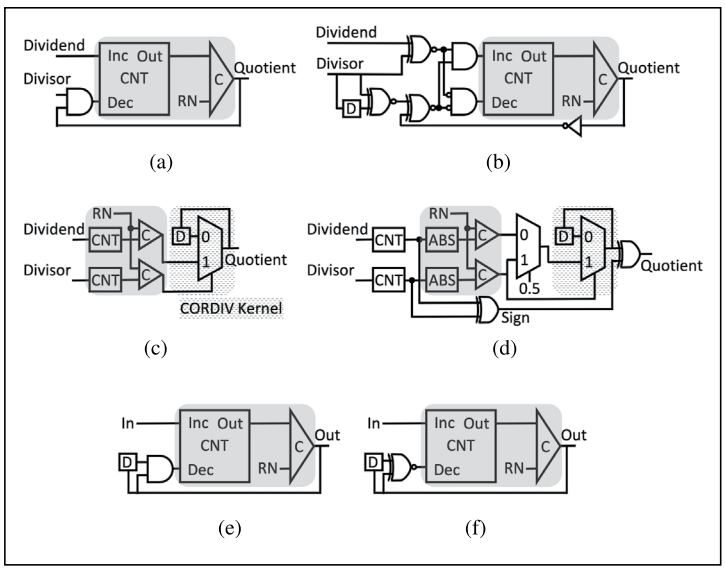

Figure 2. Existing SC division and square root. (U and B represent unipolar and bipolar, respectively; C stands for comparator; D stands for D-flip-flop; gray blocks are the BS regeneration logic; dotted blocks are CORDIV kernel.) (a) U: GDIV. (b) B: GDIV. (c) U: CORDIV. (d) B: CORDIV. (e) U: GSQRT. (f) B: GSQRT.

have researchers shown that it is possible to leverage dedicated correlation [6] to design SC division [7], [8], instead of viewing it as a detriment to accuracy. Based on the fact that emerging deep learning models employ more nonlinear operations besides linear multiplication and addition [9], our work leverages correlation to design in-stream (i.e., do not require expensive BS regeneration to achieve high accuracy) SC units (SCUs) for both division and square root, which target stochastic BSs, instead of deterministic BSs in [10], the correlation of which is ignored.

In this work, our proposed designs, in-stream correlation-based division (ISCBDIV) and bit-inserting square root (BISQRT) eliminate the fundamental inefficiencies of existing SC division and square root [1] via low-cost in-stream mechanisms. Furthermore, this work extends previous designs for unipolar SC in [8] to bipolar SC. The proposed ISCB-DIV is derived from the insight that when the dividend is larger than the divisor, the quotient saturates at 1. ISCBDIV applies a simple in-stream mechanism, skewed synchronizer (SS), to maximally correlate the dividend BS to the divisor BS of the existing non-in-stream correlated division (CORDIV) [7] for higher in-stream accuracy than others. On the other hand, BISQRT leverages the insight that the result of any SC square root is always no less than its input to insert extra 1's into the input BS to obtain the output. We introduce two low-cost mechanisms for insertion, namely stochastic insertion [8] and opportunistic insertion, thus eliminating BS regeneration in the classical Gaines design [1]. The newly designed opportunistic insertion has even higher accuracy than the previous stochastic insertion [8], without introducing overhead on top of Gaines's designs. Experiments demonstrate that the proposed designs are more suitable than the state-of-the-art competitors for emerging deep learning models.

Background

Existing designs on stochastic division and square root originate from Gaines' designs [1]. Though later CORDIV [7] improved the accuracy of SC division, there is little subsequent work on SC square root, and our work steps into this rarely explored field.

Gaines division

Gaines division (GDIV) [1] for unipolar and bipolar SC is shown in Figure 2a and b. In GDIV, a feedback loop is formed between the quotient and the divisor to construct an equilibrium between the increments and decrements (Inc and Dec in the diagram) for the depth-*N* (i.e., *N*-bit) saturating CouNTer (CNT in the diagram). Extra logic in bipolar GDIV is to stabilize the fluctuation due to varying value signs. Equilibrium for unipolar GDIV can be achieved if the increment

($P_{Dividend}$) and decrement ($P_{Divisor} \times P_{Quotient}$) are enabled with equal probabilities. Thereafter, the quotient is successfully calculated: $P_{Quotient} = P_{Dividend}/P_{Divisor}$. The computation can be accelerated by initializing the saturating counter to half of the maximum count. However, GDIV has three limitations. First, it requires the comparison between the counter value and RN, that is, BS regeneration, incurring high area overhead. Second, the GDIV precision relies on the counter depth, resulting in a high area overhead for high precision. Third, the output accuracy is determined by the multiplication accuracy via the AND/XNOR gate for unipolar/bipolar SC. Thus, the stochastic cross correlation [3] between the BSs for divisor and quotient needs to approach zero for acceptable accuracy. Stochastic cross correlation relates to the count of aligned 1's between two BSs. When the count of aligned 1's is maximized, the correlation is maximized to +1; when minimized, the correlation is also minimized to –1.

Correlated division

CORDIV utilizes maximized stochastic cross correlation between BSs for dividend and divisor [7] fed into the CORDIV kernel (Figure 2c and d). The output $P_{Quotient} = P_{Dividend}/P_{Divisor}$ equals the ratio of the one count in the BSs for dividend and divisor: $N^1_{Dividend}/N^1_{Divisor}$. It hinges on the fact that the unipolar quotient saturates at the upper bound of the legal range, that is, 1, if the dividend is larger than the divisor for unipolar SC.

The CORDIV architectures for unipolar and bipolar SC are shown in Figure 2c and d, both requiring static binary inputs stored in the counter. Bipolar CORDIV actually computes on the absolute values and converts the result to bipolar format using an XOR gate. CORDIV is not in-stream, as a counter-RNG-comparator organization is mandatory to produce the maximally correlated BSs for the CORDIV kernel, that is, whenever the dividend bit is 1, the divisor bit has to be 1. Thus, the MUX can capture all $N_{Dividend}$ bits, and output precise quotient bits when divisor bits are 1. The D-flip-flop (D-FF) records those precise quotient bits, and outputs approximate quotient bits upon logic 0's at the divisor. As such, the quotient is maximally correlated with the dividend. CORDIV has higher accuracy than GDIV due to leveraging the correlation. Still, there are two limitations for CORDIV. First, BS regeneration requires long latency to reach a stable and accurate output in the counter, further impeding the convergence of quotient BS. Second, similar to GDIV, BS regeneration in CORDIV cause hardware area and power overheads.

Gaines square root

Gaines square root (GSQRT) is the classic, yet the only, SC square root design [1] before this work. As shown in Figure 2e and f, GSQRT can be directly derived from GDIV. They differ from the decrement signal for the depth-N saturating counter: GSQRT decrements the counter based on the square of the output. Note that bipolar GSQRT does not need the stabilization logic in bipolar GDIV as legal values are always positive. At equilibrium, the counter increments (P_{In}) and decrements (P_{Out}^2) with an equal probability, resulting in square root: $P_{Out} = \sqrt{P_{In}}$. GSQRT has the same limitations as GDIV, except that stochastic auto correlation [4], the correlation of the output BS and its shifted version, now limits the accuracy.

BS generation and regeneration

Although different techniques are proposed to increase accuracy, they give rise to extra costs of latency and hardware. As BSs from BS generation [1], which are uncorrelated, flow through concatenated SCUs, the intermediate BSs might become correlated again, leading to lower accuracy and longer latency. As salvation, BS regeneration is applied to the intermediate BSs in existing works [6]. BS regeneration has a similar process to BS generation: they both compare the buffered binary data values to RNG outputs. However, BS regeneration utilized saturating counters to dynamically update BS values before comparison, while the comparison in BS generation is between static prestored values and the RNs. We name the SCUs requiring no organization of counter-RNG-comparator for BS regeneration as in-stream, SCUs, and they are more hardware-friendly in general.

In-stream correlation-based division

Our ISCBDIV is inspired by CORDIV [7], which applies an expensive counter-RNG-comparator organization to regenerate correlated BSs for the CORDIV kernel in Figure 2c and d. To mitigate such incurred overheads, an in-stream mechanism (i.e., without BS regeneration) for maximal BS correlation is proposed here.

Synchronizer [6] is a recent technique for increasing stochastic cross correlation of two input BSs by reordering the bits to favor 1–1 pairs and suppress 1–0 or 0–1 pairs. Synchronizer considers symmetric bit pairing for both input [8], ignoring the fact that only

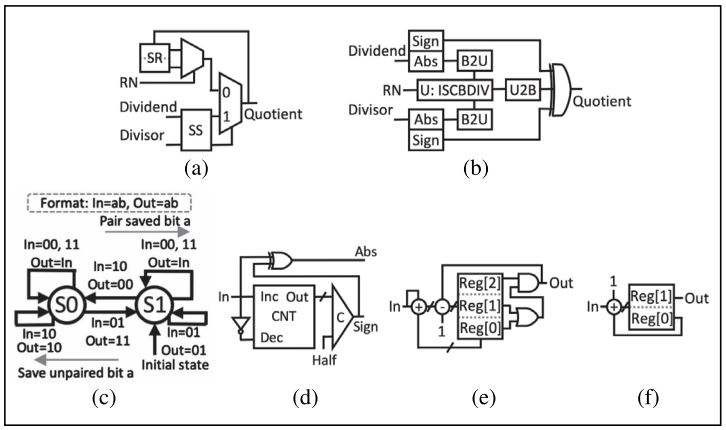

Figure 3. Proposed in-stream SC division. (a) U: ISCBDIV. (b) B: ISCBDIV. (c) SS FSM. (d) Abs/Sign. (e) B2U. (f) U2B.

the dividend needs to be synchronized to the divisor in CORDIV. Leveraging such fact, we design an SS for the maximum correlation requirement of CORDIV kernel, with the finite state machine (FSM) given in Figure 3c, where {*a, b*} represents the {*dividend, divisor*} pair. The SS employs one depth-*N* (N-bit), $N = 2$ in Figure 3c, counter to record the 1's in the dividend when the divisor bit is 0, and then matches the saved 1's with later 1's in the divisor BS [8].

The hardware architecture of the proposed unipolar ISCBDIV is presented in Figure 3a, consisting of the SS and the CORDIV kernel as in Figure 2c and d. The SS first converts input BSs with arbitrary correlation to those with maximal correlation. The resultant divisor BS remains identical, but the dividend BS is shuffled. Then the CORDIV kernel performs the actual division. The D-FF in the CORDIV kernel is replaced by a depth-*N* shift register (SR) to improve accuracy, that is, introducing higher randomness to better track the quotient value. The SR values go to the MUX and are indexed by an RN, where usually one bit satisfies, leading to depth-2 SR. ISCBDIV is significantly more hardware-efficient than CORDIV, whose overhead from regenerating BSs includes the registers to store the binary numbers, the RNG and the two comparators. For bipolar ISCBDIV in Figure 3b, we first obtain the sign and absolute value of each BS as in Figure 3d. This unit records the BS history using a saturating counter, initialized to half of the maximum, and sets the sign to 1 if the counter value is less than half, that is, more 0's than 1's. The absolute BS is simply obtained by XORing the sign BS with the original BS [7]. Then the absolute value of bipolar BS is converted to unipolar BS using bipolar-to-unipolar conversion (B2U) logic, which is a nonscaled addition performing $P_{\text{Uni}} = 2 \times P_{\text{Bi}} - 1$. Next, normal unipolar ISCBDIV is performed. The resultant unipolar quotient is converted back to bipolar with unipolar-to-bipolar conversion (U2B), a scaled addition performing $P_{\text{Bi}} = (P_{\text{Uni}} + 1)/2$. This quotient is XORed with two signs to get the final signed quotient. Note that two unipolar/bipolar interconversions maintain the BS values, but change the data polarity. To ensure the accuracy, the bipolar nonscaled and unipolar scaled adders in [2] are used for B2U in Figure 3e and U2B in Figure 3f. The B2U calculates the output based on the difference between the expected and actual output one count [2], here dictated by the register (Reg). The U2B accumulates the sum of input and 1 among multiple cycles in register, and overflows the carry bit as the output [2].

Bit-inserting square root

To alleviate the overhead of BS regeneration in GSQRT, we propose SC square root via correlation. Our BISQRT is based on the observation that the output of SC square root is always greater than or equal to the input. Thus, to produce the expected result, it is sufficient to intelligently insert 1's into the input BS. We propose two mechanisms to insert 1's properly, including stochastic insertion and opportunistic insertion. Stochastic insertion randomly replaces input bits, even input 1's, with extra 1's [8] while the opportunistic insertion merely replaces input 0's with expected 1's, leading to the maximum correlation between input and output.

Stochastic insertion mechanism

The stochastic insertion mechanism is derived from Figure 4a. The multiplexer (MUX) selectively inserts 1's according to the trace block output probability, P_{Trace}, which is formulated in (1). The output BS probability P_{Out} is represented by P_{Tmce} and Pj_n in the first line. In the second and third lines, the probability of 1's in the input BS, Pj_n, is expressed as a function of the BS value, $V_{\text{In}} = m \times Pj_n - k$, where $m = 1$, $k = 0$ for unipolar SC and $m = 2$, $k = 1$ for bipolar SC, respectively. This V_{In} is then transformed into V'_{Out} in the fourth line. Finally, in the fifth line, the equation formulates the relationship between P_{Tmce} and P_{Out}. Solve it and obtain P_{Trace} as in (2)

$$\begin{aligned} P_{\text{Out}} &= 1 \times P_{\text{Trace}} + P_{\text{In}} \times \left(1 - P_{\text{Trace}}\right) \\ &= P_{\text{Trace}} + \frac{\left(m \times P_{\text{In}} - k\right) + k}{m} \times \left(1 - P_{\text{Trace}}\right) \end{aligned}$$

$$= P_{\text{Trace}} + \frac{V_{\text{In}} + k}{m} \times \left(1 - P_{\text{Trace}}\right) \quad (1)$$

$$= P_{\text{Trace}} + \frac{V_{\text{Out}}^2 + k}{m} \times \left(1 - P_{\text{Trace}}\right)$$

$$= P_{\text{Trace}} + \frac{\left(m \times P_{\text{Out}} - k\right)^2 + k}{m} \times \left(1 - P_{\text{Trace}}\right)$$

$$P_{\text{Trace}} = \frac{\left(m \times P_{\text{Out}} - k\right)}{\left(m \times P_{\text{Out}} - k\right) + 1}. \quad (2)$$

Thus, with an MUX and a divider, we can construct stochastic BISQRT (BISQRT-S) for both unipolar and bipolar SC, and we introduce two possible architectures for stochastic BISQRT.

JKDIV-based trace block

JK-flip-flop naturally performs $P_q = P_j / (P_j + P_k)$, denoted as JK Division (JK-DIV) [1]. Thus, setting port J to $m \times P_{\text{Out}} - k$ and port K to 1 results in correct P_{Trace}. Based on this, we build BISQRT-S-JK, shown in Figure 4c, and d. The unipolar BISQRT-S-JK in Figure 4c directly connects the output to port J as $P_{\text{Uni}} = V_{\text{Uni}}$, while the bipolar version requires to send a BS of value $2 \times P_{\text{Out}} - 1$ to port J, essentially the function of B2U.

ISCBDIV-based trace block

The P_{Trace} division can be alternatively implemented with simplified ISCBDIV. This architecture, BISQRT-S-IS, is shown in Figure 4e and f. Similar to BISQRT-S-JK, bipolar implementation also has a B2U logic. BISQRT-S-IS further simplifies the SS to one AND gate and one OR gate. The dividend (MUX input port 1 in CORDIV kernel) is $(m \times P_{\text{Out}} - k)/2$, half of the output BS in unipolar SC, or the BS from the B2U in bipolar SC, achieved by an AND gate and a periodic BS of probability 0.5 generated by the D-FF and inverter. The divisor $((m \times P_{\text{Out}} - k) + 1)/2$ (MUX select port in CORDIV kernel) is generated via correlation with the simplified SS using an OR gate.

Opportunistic insertion mechanism

The opportunistic insertion mechanism is derived from Figure 4b, where extra 1's are only inserted when input bit is 0 according to the emit block. The functionality is formulated in (3), where the involved symbols are identical to those in stochastic insertion mechanism. This formulation is essentially a nonscaled addition (NS Add) for unipolar SC. The P_{In} in the first line indicates that the input BS is kept unchanged, thus

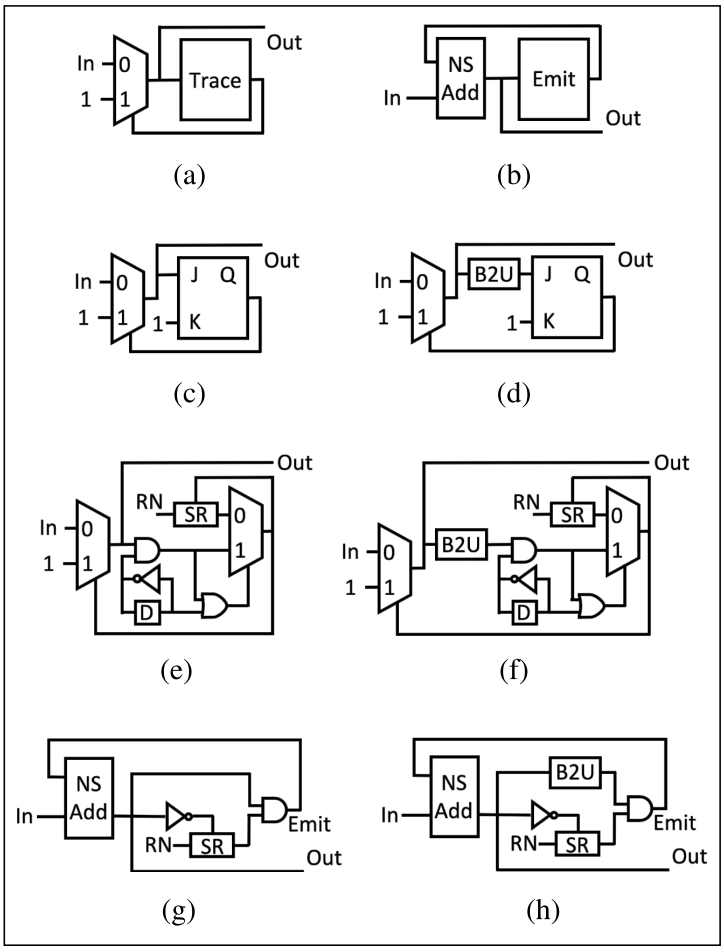

Figure 4. Proposed in-stream SC square root. (a) Stochastic insertion. (b) Opportunistic insertion. (c) U: BISQRT-S-JK. (d) B: BISQRT-S-JK.(e) U: BISQRT-S-IS. (f) B: BISQRT-S-IS. (g) U: BISQRT-O. (h) B: BISQRT-O.

input BS and output BS are maximally correlated. The P_{Emit} represents the BS of extra 1's to be emitted into the input BS. The solution for unipolar and bipolar SC is presented as in (4), implying that it can be implemented with an SC multiplication

$$P_{\text{Out}} = P_{\text{In}} + P_{\text{Emit}}$$

$$= \frac{\left(m \times P_{\text{In}} - k\right) + k}{m} + P_{\text{Emit}}$$

$$= \frac{V_{\text{In}} + k}{m} + P_{\text{Emit}}$$

$$= \frac{V_{\text{Out}}^2 + k}{m} + P_{\text{Emit}}$$

$$= \frac{\left(m \times P_{\text{Out}} - k\right)^2 + k}{m} + P_{\text{Emit}} \quad (3)$$

$$P_{\text{Emit}} = \left(m \times P_{\text{Out}} - k\right) \times \left(1 - P_{\text{Out}}\right). \quad (4)$$

Though the formulation in (4) is concise, a naive implementation will be problematic due to correlation. For unipolar SC, $P_{Emit} = P_{Out} \times (1 - P_{Out})$, where bits in the BS for $1 - P_{Out}$ are the inversion of bits in the BS for P_{Out}. Thus, the BSs for P_{Out} and $1 - P_{Out}$ have a minimal stochastic cross correlation of –1, that is, no 1–1 pair exists, leading to incorrect multiplication. Similar correlation variation will happen to bipolar SC, significantly damaging the accuracy.

To mitigate the deviation from optimal zero correlation for SC multiplication, we introduce a decorrelation approach based on SR. The unipolar and bipolar versions of the proposed opportunistic BISQRT (BISQRT-O) are depicted in Figure 4g and h, where NS Add refers to the nonscaled addition for unipolar SC. For both designs, we rerandomize the BS for $1 - P_{Out}$ with a depth-N SR, whose output is indexed by an evenly distributed RN sequence. This method is similar to the decorrelator in [6] to create near zero correlation, but only scrambles one BS instead of two as in decorrelator. Then, the scrambled BS for $1 - P_{Out}$ and the BS for $m \times P_{Out} - k$ are multiplied using an AND gate to generate an emit bit for nonscaled addition. The unipolar BS for $m \times P_{Out} - k$ is directly the BISQRT-O output, while the bipolar BS requires a B2U, similar to the situation in bipolar BISQRT-S. Lastly, the nonscaled addition also adopts the design in [2] to guarantee accuracy.

Implementation and analysis

Hardware implementation

As prior research [12] indicates that SC energy efficiency often falls short of the binary counterpart when data are beyond 8 bits, we focus on designs with 8-bit inputs and synthesize them using Synopsys Design Compiler at 400 MHz with TSMC 45-nm technology. The result is presented in Table 1. GDIV and GSQRT are both depth-5 [1]. ISCBDIV has depth-2 SS and depth-2 CORDIV kernel SR. BISQRT-O has depth-4 SR.

Accuracy simulation

All accuracy simulations are performed with our open-source UnarySim [2]. For both division and square root, the 8-bit binary input data traverse the entire legal range. For division, the dividend ranges within [0,1] for unipolar and [–1,1] for bipolar, while the divisor excludes 0. For square root, both unipolar and bipolar inputs range within [0,1]. Furthermore, to generate different input BSs, we apply 100 high-quality Sobol RNGs [5] and 100 conventional LFSR RNGs, with results on the right side of Table 1.

Overall evaluation

According to Table 1, our designs simultaneously achieve less area, less power and higher accuracy in most cases. It is important to note that all of our designs, except bipolar ISCBDIV, are effectively "plug-and-play" on most stochastic systems, since the choice of RNG has

Table 1. Implementation results with 8-bit binary inputs and RNGs, that is, 256 cycles for SC BS; thus isolatency comparison: all designs have the same throughput of 1.5625M operations per second. The best number of each column is highlighted. Notation: SS refers to the skewed synchronizer; Abs refers to the unit to retrieve the absolute value of a bipolar BS; B2U and U2B refer to the BS converters between bipolar and unipolar SC; RMSE is the root mean square error; MAE is the mean absolute error; Bias is the sum of error. All data in this table, as well as walkthrough examples, are available in UnarySim [11].

Design	Area Breakdown (%)					Area	Power	Area×Delay	Error–Sobol RNG (%)			Error–LFSR RNG (%)		
	SS	Abs	B2U	U2B	other	(μm^2)	(μW)	($\mu m^2 \cdot \mu S$)	RMSE	MAE	Bias	RMSE	MAE	Bias
U:GDIV	-	-	-	-	100	70.6	18.2	45.2	8.6	7.0	-4.5	9.8	7.8	-4.7
U:CORDIV	-	-	-	-	100	209.6	62.2	134.1	7.2	3.8	3.0	**9.2**	**4.6**	**2.2**
U: *ISCBDIV*	63.9	-	-	-	36.1	**37.9**	**14.4**	**24.3**	**4.9**	**1.9**	**-1.9**	10.8	6.2	-5.8
B:GDIV	-	-	-	-	100	**85.7**	**26.4**	**54.8**	64.5	37.8	-25.6	53.7	42.4	-1.6
B:CORDIV	-	-	-	-	100	254.0	72.0	162.6	15.8	10.6	6.9	54.7	40.1	26.2
B: *ISCBDIV*	15.8	36.1	31.1	8.0	8.9	152.8	61.4	97.8	**8.5**	**3.9**	**-0.0**	**48.4**	**34.6**	**0.0**
U:GSQRT	-	-	-	-	100	80.3	23.3	51.4	10.7	7.3	5.2	5.9	5.4	-5.1
U: *BISQRT-S-JK*	-	-	-	-	100	**11.3**	**6.3**	**7.2**	8.7	6.9	-1.9	10.8	8.8	-8.8
U: *BISQRT-S-IS*	-	-	-	-	100	23.3	12.5	14.9	10.8	8.3	-8.0	12.4	10.5	-10.5
U: *BISQRT-O*	-	-	-	-	100	52.6	19.0	33.7	**3.3**	**2.2**	**0.2**	**5.2**	**2.4**	**-1.7**
B:GSQRT	-	-	-	-	100	84.0	24.9	53.8	11.7	8.8	1.5	51.4	41.9	-41.9
B: *BISQRT-S-JK*	-	-	72.9	-	27.1	**32.3**	**15.0**	20.7	7.2	4.8	-2.8	11.4	9.9	-9.8
B: *BISQRT-S-IS*	-	-	56.6	-	43.4	40.0	19.2	25.6	7.2	4.8	-2.8	12.1	10.1	-9.4
B: *BISQRT-O*	-	-	36.6	-	63.4	74.0	28.1	47.4	**4.4**	**2.8**	**-1.2**	**10.1**	**8.3**	**-8.2**

minimal impact on the accuracy. Bipolar ISCBDIV is less "plug-and-play" due to the preceding absolute and B2U units, without which the unipolar ISCBDIV first maximizes input correlation and incurs limited accuracy loss when concatenating unipolar ISCBDIV with other SCUs.

In terms of division, CORDIV always has the largest area and power due to regenerating both dividend and divisor BSs, while GDIV costs less with BS regeneration for the only quotient. ISCBDIV totally substitutes such bottleneck components with lost-cost in-stream SS and achieves the highest accuracy for both unipolar and bipolar SC. However, bipolar ISCBDIV consumes medium area and power due to the extra absolute unit and polarity conversion units.

For square root, BISQRT-S-JK consistently has the smallest area and power due to its ultrasimple architecture. BISQRT-S-IS increases the hardware cost due to simplified ISCB-DIV. BISQRT-S has medium accuracy compared with others. Though consuming more area and power than BISQRT-S, BISQRT-O has less area and power compared to GSQRT. Moreover, it has the best accuracy of all, mitigating the correlation problem by rerandomizing the BS.

In this article, we identify the overheads in existing hardware of two nonlinear SC operations—division and square root, required by emerging deep learning models—and propose more efficient designs—ISCB-DIV and BISQRT—that leverage correlation. We evaluate the proposed designs and observe that our ISCBDIV and BISQRT achieve higher accuracy with less area and power, compared against existing work, making them better candidates for future deep learning applications. ■

References

[1] B. R. Gaines, "Stochastic computing systems," in *Advances in Information Systems Science*. New York, NY: Springer Science+Business Media, 1969.

[2] D. Wu et al., "UGEMM: Unary computing architecture for GEMM applications," in *Proc. ISCA*, May 2020, pp. 377–390.

[3] A. Alaghi and J. P. Hayes, "Exploiting correlation in stochastic circuit design," in *Proc. ICCD*, Oct. 2013, pp. 39–46.

[4] F. Neugebauer, I. Polian, and J. P. Hayes, "Building a better random number generator for stochastic computing," in *Proc. DSD*, Aug. 2017, pp. 1–8.

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

[6] V. T. Lee, A. Alaghi, and L. Ceze, "Correlation manipulating circuits for stochastic computing," in *Proc. DATE*, Mar. 2018, pp. 1417–1422.

[7] T.-H. Chen and J. P. Hayes, "Design of division circuits for stochastic computing," in *Proc. ISVLSI*, Jul. 2016, pp. 116–121.

[8] D. Wu and J. S. Miguel, "In-stream stochastic division and square root via correlation," in *Proc. DAC*, Jun. 2019, pp. 1–6.

[9] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. NeurIPS*, 2017, pp. 3856–3866.

[10] M. H. Najafi et al., "Performing stochastic computation deterministically," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 12, pp. 2925–2938, Dec. 2019.

[11] D. Wu and R. Yin. *UnarySim*. Accessed: Jan. 17, 2021. [Online]. Available: https://github.com/diwu1990/UnarySim

[12] V. T. Lee et al., "Architecture considerations for stochastic computing accelerators," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 11, pp. 2277–2289, Nov. 2018.

Di Wu is currently pursuing a PhD with the ECE Department, University of Wisconsin–Madison, Madison, WI, USA. His research interests include stochastic and approximate computing, as well as numerical optimization for deep neural networks. Wu has a master's degree in integrated circuit engineering from Fudan University, Shanghai, China. He is a member of ACM.

Ruokai Yin is currently working with Prof. Joshua San Miguel on unary computing and applying this paradigm to multiple real-world applications. His research interests include designing high-performance computer architecture for machine learning. Yin is a Senior Undergraduate with the Electrical Engineering and Computer Science Department, University of Wisconsin–Madison, Madison, WI, USA.

Joshua San Miguel is an Assistant Professor with the University of Wisconsin–Madison, Madison, WI, USA. His research interests include stochastic and approximate computing, intermittent computing and interconnection networks. San Miguel has a PhD in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada. He is a member of IEEE and ACM.

■ Direct questions and comments about this article to Di Wu, Department of Electrical and Computer Engineering, University of Wisconsin-Madison, Madison, WI 53706 USA; di.wu@ece.wisc.edu.