

CoMix-D: A Low-Cost, RNG-Free Decorrelator via Correlation Mixing for Stochastic Computing

Yexian Lin¹, Chunyan Wu¹, Kuncai Zhong^{2,*}, Weikang Qian³

¹School of Physics and Electronics, Hunan University, China, ²College of Semiconductors, Hunan University, China,

³Global College, Shanghai Jiao Tong University, China

Email: {ivylin, chunyanwu, kczhong}@hnu.edu.cn, qianwk@sjtu.edu.cn; *corresponding author

Abstract—Stochastic computing, an unconventional computing paradigm, often struggles with the costly random number generator (RNG)-based decorrelators. To solve this issue, we propose CoMix-D, a real-time solution that needs no RNGs. It uses a deterministic mixing architecture built from LiteSync, LiteDesync, and BitAggregator. Compared to state-of-the-art methods, CoMix-D achieves substantial savings of 80.1% in area and 59.9% in power without compromising accuracy.

I. INTRODUCTION

Stochastic computing (SC) [1] is a hardware-efficient computing paradigm for applications like neural networks [2]–[4], decoding of error-correction coding [5]–[7], and image processing [8]–[10]. It represents a value as the probability of a one in a bit stream, referred to as a *stochastic number* (SN) [11] (e.g., ‘11100011’ represents 5/8). Typically, an SN is generated by a stochastic number generator (SNG) [12] (Fig. 1(a)), consisting of a random number generator (RNG) to generate random binary numbers and a comparator to compare the input binary number with them.

However, achieving high computation accuracy in SC often demands low correlation between input SNs [12], [13], making decorrelators indispensable. This correlation is formally quantified by the SC correlation (SCC) [14]. The conventional “regeneration” solution (Fig. 1(b)) addresses this but suffers from prohibitive latency and hardware overhead, dominated by the area-intensive RNGs [15]. Some state-of-the-art alternatives either still require costly RNGs [16] or are non-real-time (i.e., requiring a latency period before a valid output) [17]. For further improvement, in this work, we propose CoMix-D, a novel real-time RNG-free decorrelator using lightweight finite-state machines (FSM)-based correlation manipulation and a precise bit-aggregation mixing framework.

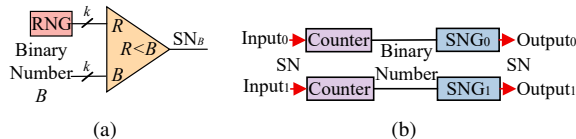


Fig. 1. Diagram of (a) an SNG and (b) the regenerate-based architecture.

This work is supported by the Natural Science Foundation of Hunan Province (No. 2025JJ60433), and the Natural Science Foundation of Changsha City (No. kq2402003).

II. PROPOSED CoMix-D DECORRELATOR

The architecture of CoMix-D is shown in Fig. 2. Our approach is based on the insight that uncorrelated SNs ($SCC = 0$) can be obtained by precisely mixing positively correlated SNs ($SCC \approx +1$) with negatively correlated ones ($SCC \approx -1$). While this concept enables the complete elimination of RNGs, its practical implementation presents three significant challenges that we solve systematically: 1) How to generate correlated SNs in real time with low hardware cost? 2) How to mathematically derive the exact mixing ratios? 3) How to physically implement the mixing ratios?

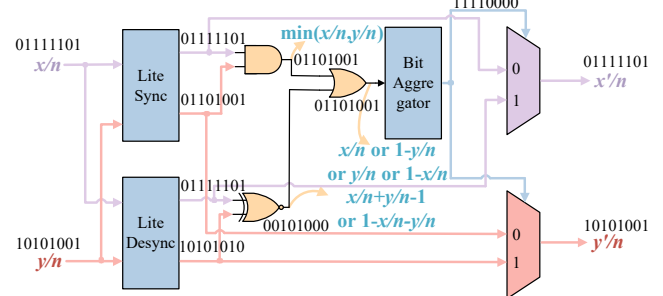


Fig. 2. The architecture of CoMix-D.

A. Low-Cost Generation of Correlated Streams

To address the first challenge, we developed LiteSync (Fig. 3(a)) and LiteDesync (Fig. 3(b)), two compact FSMs. They have a parameter D controlling the number of consecutive inputs of the same pattern that can be converted into another. For example, only one consecutive pattern (e.g., $X = 1, Y = 0$, denoted as ‘10’) can be converted to another (e.g., ‘11’) in Fig. 3(a). The initial states of the FSMs can be arbitrary. To push the SCC toward +1, we have to maximize the count of matching bit-pairs (‘11’, ‘00’) [14]. Our LiteSync handles this by turning mismatching pairs (‘10’, ‘01’) into matching ones. Conversely, LiteDesync targets a negative correlation ($SCC \approx -1$) by doing the opposite.

Unlike the earlier work [16], we reduce the state count to just two by combining the extra states in the designs in [16]. Table I shows that our designs save considerable area and power, while keeping the correlation quality at a similar level.

B. Derivation of the Correlation Mixing Ratios

The second challenge is figuring out the exact mixing ratios. To obtain the final output SNs with $SCC = 0$, we

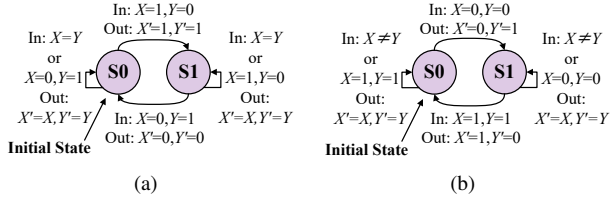


Fig. 3. Diagrams of (a) LiteSync and (b) LiteDesync with $D = 1$.

TABLE I
OUTPUT SCC AND HARDWARE COST OF LITESYNC AND LITIDESYNC
COMPARED WITH PREVIOUS WORK ($D = 1$).

Design	LiteSync	Sync. [16]	LiteDesync	Desync. [16]
Output SCC	0.9895	0.9995	-0.9894	-0.9894
Area (μm^2)	19.68	26.87	19.68	31.92
Power (μW)	3.51	5.17	4.49	5.95

need to ensure that the number of ‘11’ bit-pairs in them is $a_{\text{target}} = xy/n$. LiteSync and LiteDesync produce counts of ‘11’ as $a_{\text{max}} = \min(x, y)$ and $a_{\text{min}} = \max(x + y - n, 0)$, respectively. Since $a_{\text{min}} \leq a_{\text{target}} \leq a_{\text{max}}$, valid mixing ratios exist. We define p as the ratio of share from LiteDesync and q as the ratio of share from LiteSync (with $p + q = 1$). We can write the weighted average equation as:

$$p \cdot a_{\text{min}} + q \cdot a_{\text{max}} = xy/n. \quad (1)$$

By solving Eq. (1) for the various input cases, we can find the specific p and q values, which are the mixing ratios that we needed. Specifically, the coefficient p is determined as $p = x/n$ if $x + y \geq n$ and $x < y$, $p = y/n$ if $x + y \geq n$ and $x \geq y$, $p = 1 - y/n$ if $x + y < n$ and $x < y$, and $p = 1 - x/n$ if $x + y < n$ and $x \geq y$.

C. Implementation of the Deterministic Mixing

The last challenge is the physical implementation. Our solution is to use a multiplexer (MUX) to choose bits from LiteSync and LiteDesync outputs. This is driven by a select signal S_p set to match the value p .

First, we generate S_p using the gates in yellow in Fig. 2. It can be proven that the output probability of the OR gate equals p . However, we also face a serious correlation issue: since S_p and the data inputs of the MUX come from the same source SNs, they are correlated. Using S_p directly as the select input of the MUX would make Eq. (1) invalid.

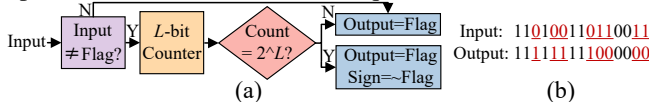


Fig. 4. Design of the BitAggregator: (a) functional architecture and (b) functional illustration with $L = 2$.

We reduce the correlation by adding the BitAggregator module (Fig. 4(a)). The main idea is to group the input bits into chunks of 1s or 0s, as exemplified by the output in Fig. 4(b). To achieve this, BitAggregator keeps a flag toggling between 0 and 1. An L -bit counter tracks the count of the bits in S_p different from the current flag. Once the count reaches a threshold of 2^L , the flag toggles. As illustrated in Fig. 4(b), this transforms an input stream frequently switching between 0 and 1 into one less frequently switching. The module effectively decorrelates the MUX’s select signal and the MUX’s input signals, ensuring the accuracy of the mixing process.

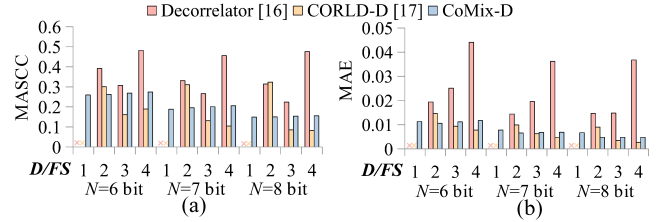


Fig. 5. Accuracy evaluation of CoMix-D with $L = 4$ under 100% correlated LD bit-stream inputs using (a) MASCC and (b) MAE.

III. EVALUATION

We evaluate CoMix-D against the state-of-the-art scrambling-based decorrelator [16] and regeneration-based CORLD-D [17]. The evaluation is conducted using low-discrepancy (LD) bit streams [18] as inputs, which are generated based on Sobol sequences [19]. We measure accuracy using mean absolute SCC (MASCC) and mean absolute error (MAE). MAE measures the average difference between the ideal product and the result of SN multiplication after a decorrelator. Meanwhile, hardware overhead is evaluated using post-synthesis area and power based on Nangate 45nm technology [20]. Note that D and FS denote the configuration parameters for the reference designs as detailed in [16], [17], whereas the parameter D for CoMix-D is defined in Section II-A.

As shown in Fig. 5, CoMix-D achieves competitive or superior accuracy (lower MASCC and MAE) compared to prior works across various bit-widths (N). Moreover, it provides substantial hardware savings, as shown in Tables II and III, where the superscripts ¹ and ² denote comparisons against the decorrelator in [16] with $N = 6$ and CORLD-D in [17], respectively. Specifically, it achieves maximum area reductions of 80.1% (AR(%)¹) and 41.7% (AR(%)²), and maximum power reductions of 48.3% (PR(%)¹) and 59.9% (PR(%)²).

TABLE II
AREA (μm^2) OF CoMix-D COMPARED WITH PREVIOUS WORK.

D/FS	[16]					[17] CoMix-D	AR (%) ¹	AR (%) ²
	$N=6$	$N=7$	$N=8$	$N=9$	$N=10$			
1	-	-	-	-	-	96	-	-
2	588	735	896	1072	1268	158	117	80.1
3	619	767	928	1133	1328	209	155	75.0
4	645	792	953	1152	1347	271	158	75.5

TABLE III
POWER (μW) OF CoMix-D COMPARED WITH PREVIOUS WORK.

D/FS	[16]					[17] CoMix-D	PR (%) ¹	PR (%) ²
	$N=6$	$N=7$	$N=8$	$N=9$	$N=10$			
1	-	-	-	-	-	6.4	-	-
2	18.0	20.5	21.6	24.1	26.5	22.1	9.3	48.3
3	19.8	22.3	23.8	22.8	25.1	26.2	11.4	42.4
4	20.1	24.3	25.4	23.3	25.3	28.7	11.5	42.8

IV. CONCLUSION

This paper introduced CoMix-D, a real-time, low-cost decorrelator that eliminates the reliance on costly RNGs in SC. Our novel correlation mixing architecture, enabled by efficient LiteSync/LiteDesync modules and the novel BitAggregator module that solves the MUX input correlation issue, achieves up to 80.1% area and 59.9% power reduction while maintaining competitive accuracy.

REFERENCES

- [1] B. R. Gaines, "Stochastic computing systems," in *Advances in Information Systems Science*. Springer, 1969, pp. 37–172.
- [2] C. F. F. et al., "Fully parallel stochastic computing hardware implementation of convolutional neural networks for edge computing applications," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 12, pp. 10 408–10 418, 2023.
- [3] X. Jia et al., "An energy-efficient bayesian neural network implementation using stochastic computing method," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 9, pp. 12 913–12 923, 2024.
- [4] A. Morán et al., "Digital implementation of radial basis function neural networks based on stochastic computing," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 13, no. 1, pp. 257–269, 2023.
- [5] J. Li et al., "Efficient stochastic polar decoder with correlated stochastic computing," *ArXiv*, vol. abs/2501.17644, 2025.
- [6] S. Hu et al., "Hybrid stochastic LDPC decoder with fully correlated stochastic computation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 9, pp. 3643–3654, 2022.
- [7] B. Yuan and K. K. Parhi, "Belief propagation decoding of polar codes using stochastic computing," in *International Symposium on Circuits and Systems*, 2016, pp. 157–160.
- [8] S. Aygun et al., "Agile simulation of stochastic computing image processing with contingency tables," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 10, pp. 3474–3478, 2023.
- [9] D. Lee and Y. Kim, "Design of a low-cost stochastic computing-based median filter for digital image processing," in *International SoC Design Conference*, 2024, pp. 298–299.
- [10] N. Temenos and P. P. Sotiriadis, "Modeling a stochastic computing nonscaling adder and its application in image sharpening," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 5, pp. 2543–2547, 2022.
- [11] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," *ACM Transactions on Embedded Computing Systems*, vol. 12, pp. 92:1–92:19, 2013.
- [12] A. Alaghi et al., "The promise and challenge of stochastic computing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 8, pp. 1515–1531, 2018.
- [13] A. Alaghi et al., *Accuracy and Correlation in Stochastic Computing*. Springer International Publishing, 2019, pp. 77–102.
- [14] A. Alaghi and J. P. Hayes, "Exploiting correlation in stochastic circuit design," in *International Conference on Computer Design*, 2013, pp. 39–46.
- [15] W. Qian et al., "An architecture for fault-tolerant computation with stochastic logic," *IEEE Transactions on Computers*, vol. 60, no. 1, pp. 93–105, 2011.
- [16] V. T. Lee et al., "Correlation manipulating circuits for stochastic computing," in *Design, Automation & Test in Europe Conference & Exhibition*, 2018, pp. 1417–1422.
- [17] S. Asadi et al., "CORLD: In-stream correlation manipulation for low-discrepancy stochastic computing," in *International Conference on Computer-Aided Design*, 2021, pp. 1–9.
- [18] A. Alaghi and J. P. Hayes, "Fast and accurate computation using stochastic circuits," in *Design, Automation & Test in Europe Conference & Exhibition*, 2014, pp. 1–4.
- [19] S. Liu and J. Han, "Energy efficient stochastic computing with sobol sequences," in *Design, Automation & Test in Europe Conference & Exhibition*, 2017, pp. 650–653.
- [20] Nangate Inc., <http://www.nangate.com>, 2021.