Absolute Subtraction and Division Circuits Using Uncorrelated Random Bitstreams in Stochastic Computing

Yuancheng Zhou
School of Microelectronics
Hefei University of Technology
Hefei, China
yczhou@mail.hfut.edu.cn

Guangjun Xie
School of Microelectronics
Hefei University of Technology
Hefei, China
gjxie8005@hfut.edu.cn

Jie Han
Department of Electrical and Computer Engineering
University of Alberta
Edmonton, Canada
jhan8@ualberta.ca

Yongqiang Zhang
School of Microelectronics
Hefei University of Technology
Hefei, China
ahzhangyq@hfut.edu.cn

***Abstract*—Different from conventional deterministic binary computing, stochastic computing (SC) utilizes random binary bitstreams to implement arithmetic functions. It has shown advantages in hardware cost and fault tolerance in applications such as image processing. In contrast stretching and edge detection, specifically, division and absolute subtraction are important functions. However, it is challenging to directly compute these functions in SC, especially when uncorrelated bitstreams are used. In this paper, a counter-based unipolar scaled absolute subtractor (UCASub) is first proposed for using two uncorrelated bitstreams. Based on the UCASub, a bipolar scaled absolute subtractor and unipolar and bipolar dividers are further proposed for using uncorrelated bitstreams. Experimental results show that these circuits are more accurate with lower mean squared errors and similar hardware overhead when compared with previous designs.**

***Index Terms*—Stochastic computing, absolute subtractor, divider, uncorrelated bitstreams.**

I. INTRODUCTION

As a distinctive computing technique, stochastic computing (SC) has attracted increasing attention in recent years. The characteristics of SC lie in the representation of binary numbers by the probabilities of 1s occurring in random bitstreams, instead of binary radix representations [1]. For example, a bitstream $X = 00100101$ encodes the number 3/8 and $Y = 10100001$ also encodes 3/8. This method brings two advantages to SC. First, simple logic gates can be used to realize complex functions (Fig. 1). For example, an AND gate performs multiplication. Second, since the weight of each bit in a bitstream is identical, SC has a high fault tolerance to errors induced by bit flips [2, 3].

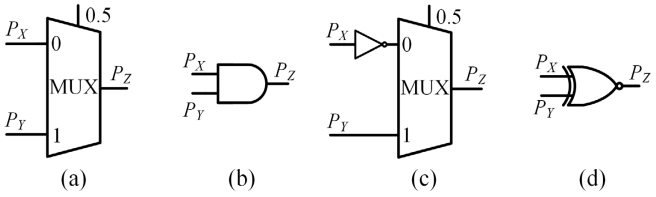

Fig. 1. Basic stochastic components. (a) Unipolar and bipolar scaled adder. (b) Unipolar multiplier. (c) Bipolar scaled subtractor. (d) Bipolar multiplier.

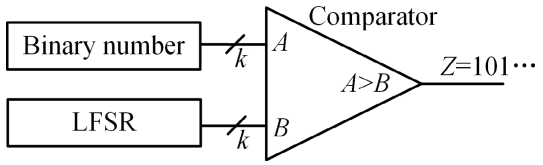

Fig. 2. A stochastic number generator (SNG).

Generally, the random bitstreams are generated by stochastic number generators (SNGs), as shown in Fig. 2. A binary number n is compared with the pseudo-random numbers produced by a k-bit linear feedback shift register (LFSR) at a rate of one bit per clock cycle. In each clock cycle, the probability of generating a 1 is approximately $n/2^k$. The value encoded by a generated bitstream is determined by the number, n, and the length of the bitstream, k. The higher the required accuracy, the longer a bitstream becomes, which in turn leads to a longer latency in computing [4]. There are two formats of bitstreams in SC: the unipolar and bipolar representations [5]. A number x in the unipolar format represented by a bitstream X is equal to the probability of 1s occurring in X, denoted by P_X here, i.e., $x = P_X$. In the bipolar format, the number x

978-1-6654-0959-9/21/$31.00 @2021 IEEE

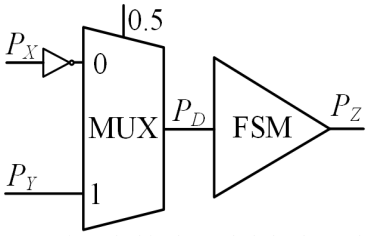

Fig. 3. The implementation of a bipolar scaled absolute subtractor [11].

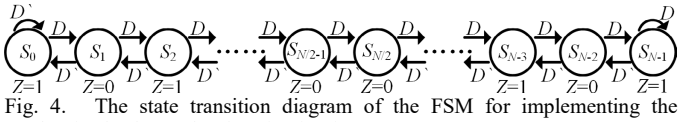

Fig. 4. The state transition diagram of the FSM for implementing the stochastic absolute value function in Fig. 3.

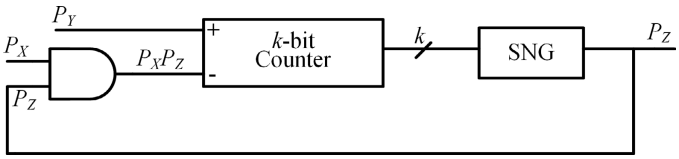

Fig. 5. The implementation of the ADDIE-based unipolar stochastic divider.

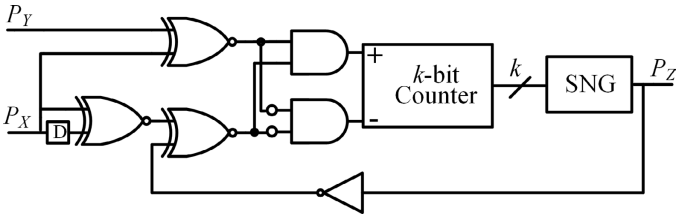

Fig. 6. The implementation of the ADDIE-based bipolar stochastic divider.

represented by the bitstream X is $2P_X-1$, i.e., $x=2P_X-1$. For instance, if X is "00101010", $x=3/8$ in the unipolar format and $x=-2/8$ in the bipolar format.

Stochastic computing correlation (SCC), including positive and negative correlations, can be exploited to realize some unique functions [6]. For example, when the input bitstreams are maximally positively correlated, an XOR gate performs the absolute subtraction in the unipolar format (UASub). However, this delicate correlation between bitstreams is hard to maintain and may be lost from one block to another in a large system [7]. Furthermore, correlations between bitstreams may affect the accuracy of some SC functions [8]. For instance, SC unipolar multiplication needs uncorrelated input bitstreams to correctly function. If they are maximally positively correlated, the result of the multiplication will mistakenly become the minimum of the inputs.

SC has successfully been applied to the fields that need many computing resources, such as image processing [6], neural networks [9, 10, 11], and digital filters [12]. In applications such as edge detection and contrast stretching, the implementations of the absolute subtraction and division are indispensable. Li *et al.* have proposed a bipolar scaled absolute subtractor (BASub) [13], in which a multiplexer implements the subtraction and a finite state machine (FSM) realizes the absolute function [14]. However, a unipolar absolute subtraction using uncorrelated bitstreams has not been studied. In addition, the unipolar and bipolar stochastic dividers, called ADDIE-based dividers, using uncorrelated bitstreams were first proposed in [1]. Those dividers are implemented using an up-down counter with a feedback path. This design has the disadvantage that it takes a long convergence time to reach an approximate result [15].

In this paper, a counter-based unipolar scaled absolute subtractor (UCASub) is proposed for using uncorrelated bitstreams. A bipolar scaled absolute subtractor and unipolar and bipolar dividers are designed by utilizing the UCASub and uncorrelated bitstreams. Experimental results show that the proposed architectures are more accurate than previous designs with similar hardware overhead.

The rest of this paper is organized as follows. The related work is reviewed in Section II. Section III presents the four proposed stochastic circuits. Section IV reports the experimental results. Section V concludes this paper.

II. Related Work

A. An FSM-Based Bipolar Scaled Absolute Subtractor

The architecture of the FSM-based bipolar scaled absolute subtractor [11] is shown in Fig. 3, where P_X, P_Y, and P_Z are the probabilities of the input/output bitstreams, and P_D is the probability of an internal signal. The MUX implements the bipolar scaled subtraction $2P_D-1=0.5((2P_X-1)-(2P_Y-1))$, and the FSM realizes the absolute value function. The state transition diagram of the FSM is shown in Fig. 4, where the output Z is determined by the current state $S_i(0\le i\le N-1)$ [14]. If $0\le i\le N/2$, the output is 1 when i is even; otherwise 0. If $N/2\le i\le N-1$, the output is 1 when i is odd; otherwise 0. The approximate function of the FSM is $2P_Z-1=|2P_D-1|$. Hence, $2P_Z-1=0.5|(2P_X-1)-(2P_Y-1)|$, so it implements the bipolar scaled absolute subtraction.

B. An ADDIE-Based Unipolar Divider

The architecture of the ADDIE-based unipolar divider (UDiv) is shown in Fig. 5 [16]. The probabilities of the input/output bitstreams are P_X, P_Y, and P_Z, respectively. Notice that the numbers in the unipolar format range in [0,1], so the divisor P_X should be larger than 0 and greater than or equal to the dividend P_Y. A k-bit up-down counter is connected to an SNG, so the binary values in the counter can be converted to bitstreams. There is a feedback loop from the output P_Z to be multiplied with P_X. If $P_Y>P_XP_Z$, the counter increases by 1. If $P_Y<P_XP_Z$, the counter decreases by 1. In other cases, the value in the counter remains unchanged. If the system is in equilibrium, $P_XP_Z=P_Y$, i.e., $P_Z=P_Y/P_X$, so the unipolar division is implemented [1].

C. An ADDIE-Based Bipolar Divider

The numbers in the bipolar format range in [-1,1]. To design

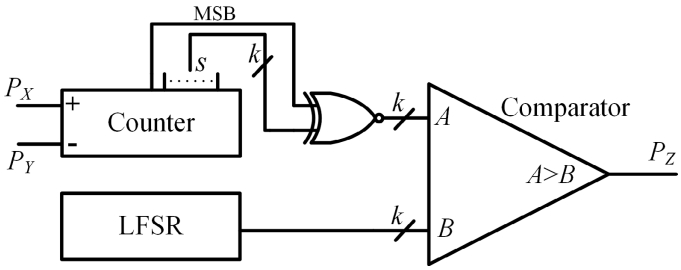

Fig. 10. The proposed counter-based unipolar scaled absolute subtractor (UCASub).

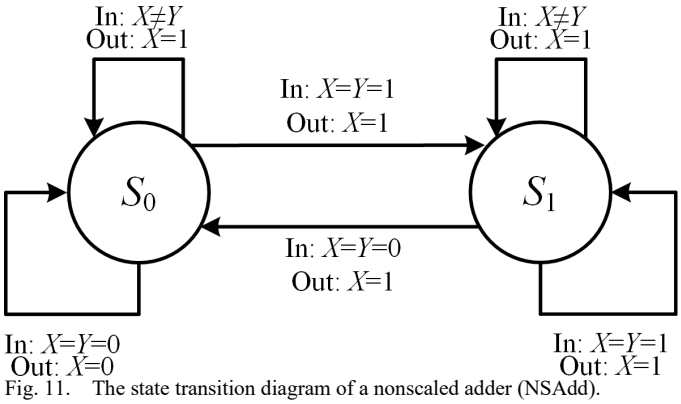

Fig. 11. The state transition diagram of a nonscaled adder (NSAdd).

a bipolar divider (BDiv) that computes $2P_Z-1=(2P_Y-1)/(2P_X-1)$, $|2P_X-1|$ needs to be larger than 0 and greater than or equal to $|2P_Y-1|$. This will result in strict restrictions on the relation between P_Y and P_X, which is summarized as follows. If $P_Y \geq 0.5$ and $P_X > 0.5$, then $P_Y \leq P_X$. If $P_Y < 0.5$ and $P_X > 0.5$, then $P_X + P_Y \geq 1$. If $P_Y < 0.5$ and $P_X < 0.5$, then $P_Y > P_X$. If $P_Y \geq 0.5$ and $P_X < 0.5$, then $P_X + P_Y \leq 1$.

The architecture of the ADDIE-based bipolar divider is shown in Fig. 6 [1]. The ADDIE-based bipolar divider similarly uses a feedback loop and an up-down counter to realize the division. The numbers in the bipolar format can be negative, so $2P_Y-1=(2P_Z-1)(2P_X-1)$ does not guarantee $2P_Z-1=(2P_Y-1)/(2P_X-1)$. Thus, the control signal of the counter is changed from $2P_Y-1$ and $(2P_Z-1)(2P_X-1)$ to $(2P_X-1)$ $(2P_Y-1)$ and $(2P_Z-1)(2P_X-1)^2$, respectively. When the system is in equilibrium [17], $(2P_X-1)(2P_Y-1)=(2P_Z-1)$ $(2P_X-1)^2$, that is $2P_Z-1=(2P_Y-1)/(2P_X-1)$, so the bipolar division is realized.

III. Proposed Designs

A. A Counter-Based Unipolar Scaled Absolute Subtractor

Implementing the absolute value subtraction between two uncorrelated bitstreams is a challenging task because the 0s and 1s in those bitstreams are randomly distributed. A counter-based unipolar scaled absolute subtractor (UCASub) is proposed to use an LFSR, a counter, and a comparator, as shown in Fig. 7. P_X and P_Y are the probabilities of the input bitstreams which are connected to the Up and Down pins of the counter respectively. The counter increases by 1 when X=1 and Y=0, and decreases by 1 when X=0 and Y=1. In other cases, the counter remains unchanged. The most significant bit (MSB) of the counter is connected to an input of an XNOR gate, and the other input is connected to one of the remaining outputs of the counter. The outputs of the XNOR gates then serve as an input to the comparator.

Assume the length of the input bitstreams is 2^k, then a k+1-bit counter and a k-bit LFSR are needed, with the initial value of the counter set to 2^k. For example, when computing the absolute value between two uncorrelated bitstreams with a length of 256 bits, a 9-bit counter and an 8-bit LFSR are needed, with the initial value of the counter set to 256. The UCASub can compute the function $P_Z = 0.5|P_X - P_Y|$.

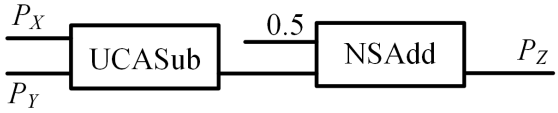

Fig. 7. The proposed UCASub-based bipolar scaled absolute subtractor.

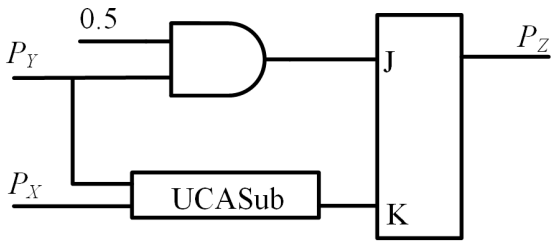

Fig. 8. The proposed UCASub-based unipolar divider.

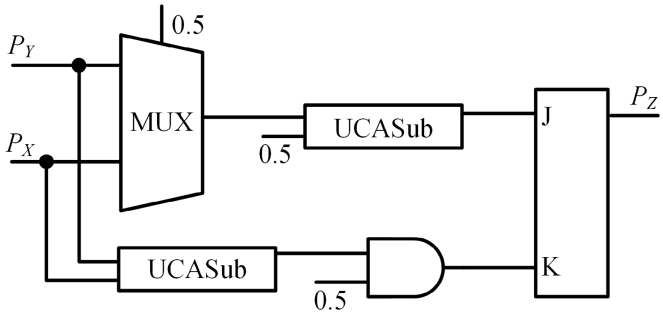

Fig. 9. The proposed UCASub-based bipolar divider.

B. An UCASub-Based Bipolar Scaled Absolute Subtractor

A new nonscaled adder (NSAdd) is proposed first. For the general unipolar addition, there is a scaling operation to guarantee that the results are in the unit interval [0,1]. If the sum of inputs is not greater than 1, the scaling operation can be

TABLE I
THE MSE OF THE UNIPOLAR SCALED ABSOLUTE SUBTRACTION FOR DIFFERENT VALUES OF P_X USING 1024-BIT STREAMS

P_X	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
MSE ($\times 10^{-2}$)	0.0053	0.0065	0.0069	0.0065	0.0077	0.0058	0.0069	0.0084	0.0076	0.0067	0.0051

TABLE II
THE MSE OF THE BIPOLAR SCALED ABSOLUTE SUBTRACTION FOR DIFFERENT VALUES OF P_X USING 256-BIT STREAMS

P_X	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
UCASub-based	0.0142	0.0067	0.004	0.0027	0.0019	0.0014	0.0019	0.0024	0.0041	0.0071	0.0129
FSM-based	0.0167	0.0157	0.0156	0.0155	0.0154	0.0152	0.0152	0.0149	0.0148	0.0148	0.0149

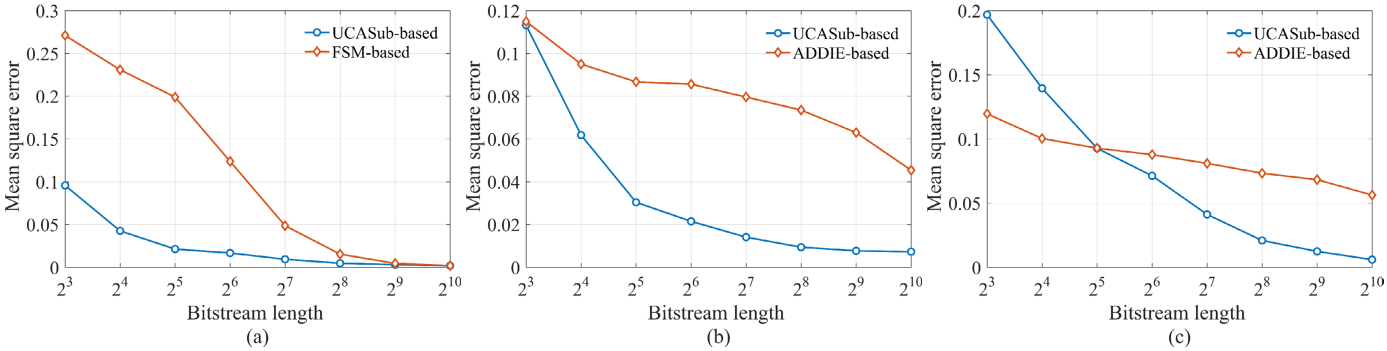

Fig. 12. Mean squared error for (a) the UCASub-based and FSM-based bipolar subtractors, (b) the UCASub-based and ADDIE-based unipolar dividers, and (c) the UCASub-based and ADDIE-based bipolar dividers.

omitted [16]. Given two bitstreams X and Y encoding probabilities P_X and P_Y. The NSAdd produces a nonscaled output encoding P_X+P_Y. The state transition diagram of the NSAdd is shown in Fig. 8. When the input bits are both 1s, one of the 1s is saved and the other is output. When both inputs are 0, the output is 1 if there are 1s that have been saved; otherwise 0. If the inputs are different, the NSAdd simply outputs the sum, so all of the 1s in the inputs are recorded.

The design of the proposed UCASub-based bipolar scaled absolute subtractor is shown in Fig. 9, where the output of the UCASub, $0.5|P_X - P_Y|$, is connected to the NSAdd. Since $0.5|P_X - P_Y| \le 0.5$, we have

$$P_Z = 0.5|P_X - P_Y| + 0.5 . \quad (1)$$

Thus,

$$2P_Z - 1 = 0.5|(2P_X - 1) - (2P_Y - 1)|, \quad (2)$$

so the bipolar scaled absolute subtraction is realized.

C. An UCASub-Based Unipolar Divider

The design of the proposed UCASub-based unipolar divider is shown in Fig. 10, where P_Y is the dividend and P_X is the divisor. The JK flip flop implements the function $P_Z = P_J/(P_J + P_K)$ as an approximate division as long as P_J is much smaller than P_K [1]. The output of the AND gate is connected to the port J and the output of the UCASub is connected to the port K. Accordingly,

$$P_J = 0.5P_Y , \quad (3)$$

$$P_K = 0.5|P_X - P_Y| = 0.5(P_X - P_Y) . \quad (4)$$

Thus,

$$P_Z = P_J/(P_J + P_K) = P_Y/P_X , \quad (5)$$

so the unipolar division is realized.

D. An UCASub-Based Bipolar Divider

The design of the proposed UCASub-based bipolar divider is shown in Fig. 11, where P_Y is the dividend and P_X is the divisor. Thus,

$$P_J = 0.5|0.5(P_Y + P_X) - 0.5| , \quad (6)$$

$$P_K = 0.25|P_X - P_Y| . \quad (7)$$

Due to the aforementioned restrictions (see Section II), the output P_Z of the JK flip flop is related with P_X and P_Y as follows. If $P_Y \ge 0.5$ and $P_X > 0.5$, or $P_Y < 0.5$ and $P_X > 0.5$, then

$$P_J = 0.5(0.5(P_X + P_Y) - 0.5) , \quad (8)$$

$$P_K = 0.25(P_X - P_Y) . \quad (9)$$

Thus,

$$2P_Z - 1 = (2P_Y - 1)/(2P_X - 1) . \quad (10)$$

If $P_Y < 0.5$ and $P_X < 0.5$, or $P_Y \ge 0.5$ and $P_X < 0.5$, then

$$P_J = 0.5(0.5 - 0.5(P_X + P_Y)) , \quad (11)$$

$$P_K = 0.25(P_Y - P_X) . \quad (12)$$

Thus,

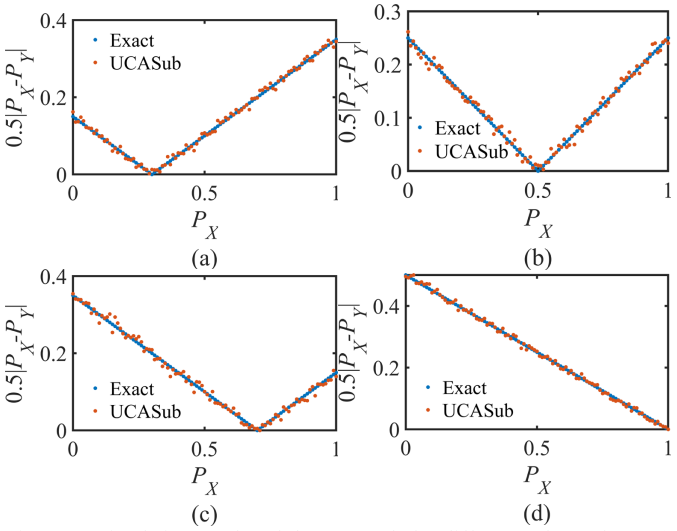

Fig. 13. Simulation results of the UCASub for different values of P_Y. (a) P_Y=0.3. (b) P_Y=0.5. (c) P_Y=0.7. (d) P_Y=1.0.

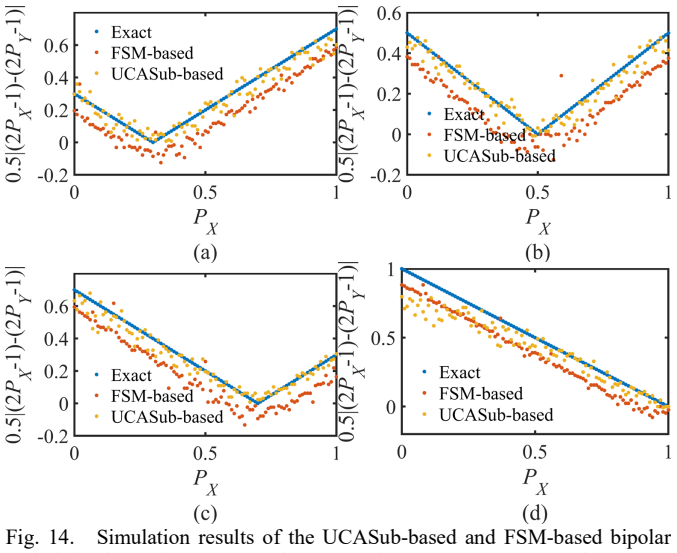

Fig. 14. Simulation results of the UCASub-based and FSM-based bipolar scaled absolute subtractors for different values of P_Y. (a) P_Y=0.3. (b) P_Y=0.5. (c) P_Y=0.7. (d) P_Y=1.0.

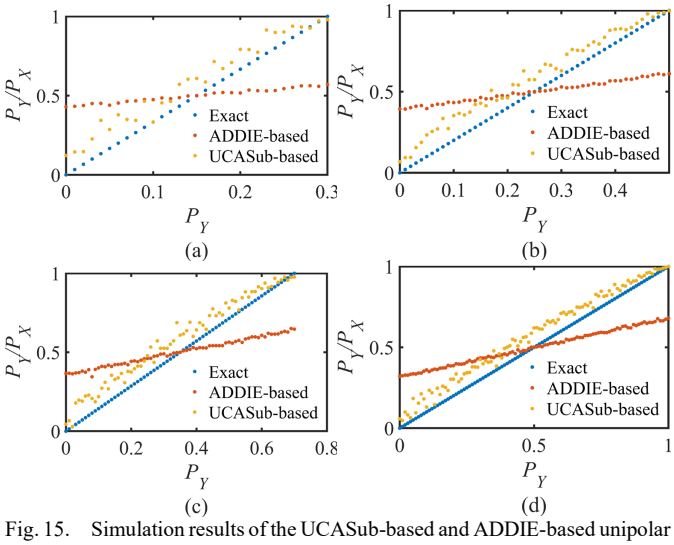

Fig. 15. Simulation results of the UCASub-based and ADDIE-based unipolar dividers for different values of P_X. (a) P_X=0.3. (b) P_X=0.5. (c) P_X=0.7. (d) P_X=1.0.

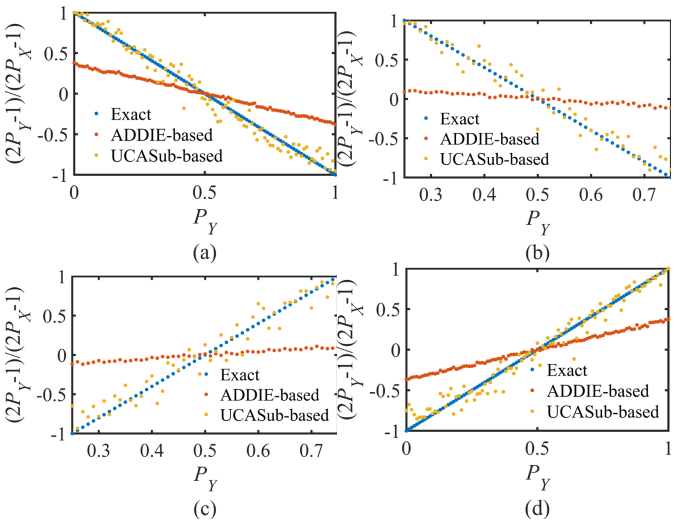

Fig. 16. Simulation results of the UCASub-based and ADDIE-based bipolar dividers for different values of P_X. (a) P_X=0. (b) P_X=0.25. (c) P_X=0.75. (d) P_X=1.0.

$$2P_Z - 1 = (2P_Y - 1) / (2P_X - 1) . \quad (13)$$

In conclusion, the output P_Z returns the result of the bipolar division.

IV. Experimental Results

A. Accuracy Comparison

Firstly, the accuracy of the proposed UCASub is measured by using the mean squared error (MSE). TABLE I lists the MSE of the UCASub in percentage for different values of P_X by randomly sampling P_Y in the unit interval [0,1]. It shows that the accuracy of the UCASub is quite steady for those values of P_X. Consider the unipolar scaled absolute subtraction $P_Z = 0.5 | P_X - P_Y |$. The value of P_X is given by 0:0.01:1, and the value of P_Y is given by 0.3, 0.5, 0.7, and 1.0, respectively. Fig. 13 shows the simulation results of the UCASub with 1024-bit streams. The results generated by the UCASub are close to the exact results.

Secondly, the accuracy comparison is made between the UCASub-based and FSM-based bipolar scaled absolute subtractors, the UCASub-based and ADDIE-based unipolar dividers, and the UCASub-based and ADDIE-based bipolar dividers. One thousand pairs of P_X and P_Y values are randomly sampled in the unit interval [0,1]. The experiment is repeated for different bitstream lengths. Fig. 12 shows that the proposed designs have lower MSEs than the other designs. In addition,

TABLE III
THE CIRCUIT AREA, POWER, AND DELAY FOR DIFFERENT DESIGNS

Function	Design	Area (μm^2)	Power (μW)	Delay (ns)
BASub	FSM-based	88.20	4.62	0.47
	UCASub-based	138.30	10.14	0.89
UDiv	ADDIE-based	105.13	6.84	1.07
	UCASub-based	123.66	9.11	1.09
BDiv	ADDIE-based	119.25	7.90	1.16
	UCASub-based	240.79	16.44	1.22

TABLE II presents a detailed comparison between the UCASub-based and FSM-based bipolar scaled absolute subtractors for different values of P_X by randomly sampling P_Y in the unit interval [0,1]. It shows that the MSE of the UCASub-based bipolar scaled absolute subtractor is always lower than that of the FSM-based one for those values of P_X.

Lastly, simulations are performed to assess the accuracy of the aforementioned designs. The bitstream length is 1024 and the sampling interval is 0.01, as shown in Fig. 14, Fig. 15, and Fig. 16. It can be seen that the proposed designs generate results that are closer to the exact results than the other designs. A longer convergence time is required for the ADDIE-based dividers, resulting in a longer bitstream length. The length of the bitstreams used in the simulations are the same so that the ADDIE-based dividers have not reached the convergence. This leads to a significant deviation between the ADDIE-based results and the exact results.

B. Hardware Evaluation

TABLE III shows the comparisons of the hardware usage between the proposed designs and the other ones by using the Synopsys Design Compiler with TSMC's 40 nm library. The first column in TABLE III lists the aforementioned three functions. Note that the delay in this table is the critical path delay. Although the delays of the proposed dividers are slightly higher than those of their counterparts, the dividers achieve lower MSEs using the same bitstream length. Hence, the computing latency of the ADDIE-based dividers is much longer than that of the proposed UCASub-based dividers for obtaining the results with the same accuracy.

V. CONCLUSION

In this paper, a counter-based unipolar scaled absolute subtractor (UCASub) is proposed to offer a new way to compute the absolute difference between two uncorrelated bitstreams. A bipolar scaled absolute subtractor and unipolar and bipolar dividers are then designed with improved performance using uncorrelated bitstreams. Simulation results show that the proposed architectures can achieve lower MSEs than previous designs at a cost of a marginal increase in hardware. Future work will focus on optimizing these designs and exploring their applications.

REFERENCES

[1] B. Gaines, "Stochastic computing systems," *Advances in information systems science*, Advances in information systems science J. T. Tou, ed., pp. 37-172: Springer, Boston, MA, 1969.

[2] P. Li, D. Lilja, W. Qian, M. Riedel, and K. Bazargan, "Logical computation on stochastic bit streams with linear finite-state machines," *IEEE Trans. Comput.,* vol. 63, no. 6, pp. 1473-1485, Jun. 2014.

[3] F. Neugebauer, I. Polian, and J. Hayes, "On the maximum function in stochastic computing," *in the Proceedings of the 16th ACM International Conference on Computing Frontiers*, pp. 59-66, 2019.

[4] S. Liu, and J. Han, "Toward energy-efficient stochastic circuits using parallel Sobol sequences," *IEEE Trans. Very Large Scale Integr. VLSI Syst.,* vol. 26, no. 7, pp. 1326-1339, Jul. 2018.

[5] A. Alaghi, and J. Hayes, "Exploiting correlation in stochastic circuit design," in the 2013 IEEE 31st International Conference on Computer Design (ICCD), Asheville, NC, USA, 2013, pp. 39-46.

[6] A. Alaghi, L. Cheng, and J. Hayes, "Stochastic circuits for real-time image-processing applications," in the 2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC), Austin, TX, USA, 2013, pp. 1-6.

[7] H. Abdellatef, M. Khalil-Hani, and N. Shaikh-Husin, "Accurate and compact stochastic computations by exploiting correlation," *Turk. J Electr. Eng. Comput. Sci.,* vol. 27, no. 1, pp. 547-564, 2019.

[8] M. Najafi, and M. Salehi, "A fast fault-tolerant architecture for sauvola local image thresholding algorithm using stochastic computing," *IEEE Trans. Very Large Scale Integr. VLSI Syst.,* vol. 24, no. 2, pp. 808-812, Feb. 2016.

[9] A. Ardakani, F. Leduc-Primeau, N. Onizawa, T. Hanyu, and W. Gross, "VLSI implementation of deep neural network using integral stochastic computing," *IEEE Trans. Very Large Scale Integr. VLSI Syst.,* vol. 25, no. 10, pp. 2688-2699, Oct. 2017.

[10] V. Canals, A. Morro, A. Oliver, M. Alomar, and J. Rossello, "A new stochastic computing methodology for efficient neural network implementation," *IEEE Trans. Neural Netw. Learn. Syst.,* vol. 27, no. 3, pp. 551-64, Mar. 2016.

[11] Y. Liu, S. Liu, Y. Wang, F. Lombardi, and J. Han, "A survey of stochastic computing neural networks for machine learning applications," *IEEE Trans. Neural Netw. Learn. Syst.,* vol. 32, no. 7, pp. 2809 - 2824, Jul. 2021.

[12] H. Ichihara, T. Sugino, S. Ishii, T. Iwagaki, and T. Inoue, "Compact and accurate digital filters based on stochastic computing," *IEEE Trans. Emerging Top. Comput.,* vol. 7, no. 1, pp. 31-43, Mar. 2019.

[13] P. Li, D. Lilja, W. Qian, K. Bazargan, and M. Riedel, "Computation on stochastic bit streams digital image processing case studies," *IEEE Trans. Very Large Scale Integr. VLSI Syst.,* vol. 22, no. 3, pp. 449-462, Mar. 2014.

[14] P. Li, and J. Lilja, "A low power fault-tolerance architecture for the kernel density estimation based image segmentation algorithm," in the IEEE International Conference on Application-specific Systems, Architectures and Processors, Santa Monica, CA, USA, 2011, pp. 161-168.

[15] S. I. Chu, "New divider design for stochastic computing," *IEEE Trans. Circuits Syst. II Express Briefs,* vol. 67, no. 1, pp. 147-151, Jan. 2020.

[16] B. Gaines, "Stochastic computing," in the Proceedings of the April 18-20, 1967, spring joint computer conference on - AFIPS '67 (Spring), Atlantic City, New Jersey, 1967, pp. 149-156.

[17] T. Chen, and J. Hayes, "Design of division circuits for stochastic computing," in the 2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Pittsburgh, PA, USA, 2016, pp. 116-121.