Highly Accurate and Energy Efficient Stochastic Multipliers

Yongqiang Zhang, *Member, IEEE,* Lingyun Xie, Jie Han, *Senior Member, IEEE,* and Guangjun Xie

Abstract—**Stochastic computing (SC) has gained interest by virtue of its small area and low power in computing units through encoding numerical values into randomly distributed bitstreams. However, this method not only results in long latency and thus increases energy consumption, but also reduces computing accuracy. Fortunately, stochastic bitstreams can be generated in parallel to reduce the latency. In this paper, parallel and serial multipliers using optimized encoding sequences with high accuracy and low energy consumption are proposed. Experimental results show that the mean squared error (MSE) is only 0.0015 for a 3-bit serial/parallel multiplier. The MSEs of 3- to 8-bit multipliers are reduced by approximately 29% on average compared with existing designs. The serial multiplier reduces the area and energy consumption by approximately 28% and 24% on average. The parallel multiplier reduces the latency and energy consumption by about 78% and 57% on average, respectively. To evaluate the effectiveness of the proposed multipliers, an image multiplication algorithm is implemented. The results show that the proposed multipliers yield higher image qualities than previous designs.**

Keywords—Stochastic computing, thermometer code, multiplier, image multiplication.

I. Introduction

Stochastic computing (SC) encodes a binary value within the interval [0, 1] into a randomly distributed bitstream. The value is represented as the proportion of the number of ones in the bitstream to the length of the bitstream. The main advantage of performing logical operations through bitstreams is that the area and power of arithmetic circuits can be significantly reduced. For example, an AND gate can implement multiplication in SC. For this reason, it is suitable for applications containing a large number of multiplication units, such as filters [1], neural networks (NNs) [2], and image processing [3].

Although the classical serial SC design requires only one AND gate to implement the multiplication, as shown in Fig. 1(a), its latency increases exponentially with the bit length n of inputs, which greatly increases the energy consumption. The area advantage is also offset to some extent by the presence of stochastic number generators (SNGs) for generating bitstreams. Although the random number source (such as the linear feedback shift register (LFSR)) in an SNG can be shared to reduce the area, the long latency cannot be alleviated by this method [4].

To reduce the latency, it is helpful to utilize the parallel thermometer code for multipliers, since all 1s in the thermometer code proceed 0s, and the 0s involved in multiplication generate 0s and do not affect the results. The generated results will remain constant after the last 1 in the thermometer code is involved in the multiplication [5], as shown in Fig. 1(b). Thus, when performing multiplication, it does not take extra time to operate on the remaining part in the thermometer code containing 0s. Using this method, a cost-effective SC multiplier has been proposed by Sim and Lee [6]. 'Sim' will be used to represent the design and generated bitstreams in [6] for easy description. It uses a finite state machine (FSM) as the multiplexer's select input to generate a relatively uniformly distributed low discrepancy sequence as the bitstream of input multiplicator. The multiplicand enters a down-counter. The down-counter is reduced by 1 per clock cycle, and the corresponding bit in the bitstream is detected at the same time. If the bit is 1, the counter's output will increase by 1; otherwise, the output will not change. When the value in the down-counter is 0, the result is ready. It has the advantage that not only does it have a high accuracy, but it no longer takes the 2^n clock cycles required by conventional SC (where n is the number of bits in the multiplier). However, if the multiplicand in the down-counter is large or even close to 1, it requires the same number of clock cycles as the classical serial stochastic multiplier.

Therefore, it becomes important to use parallel stochastic bitstreams to perform operations. A parallel bitstream generator (PBG) that produces the thermometer code in one clock cycle has been proposed in [7]. To guarantee computing accuracy, a deterministic method has been utilized in this PBG [8]. However, the deterministic method changes the bitstream length from 2^n to 2^{2n}. As shown in Fig. 2, the multiplication of just two bits uses 9 AND gates. This causes the number of AND gates in the circuit to increase tremendously even with a slight increase in the number of input bits, resulting in very high energy consumption.

In order to balance computing accuracy and energy consumption at the same time, therefore, a serial multiplier and a parallel multiplier using optimized encoding sequences are proposed in this paper. Experimental results show that the optimized encoding sequences used in the proposed multipliers lead to smaller errors than Sim's coding sequences for multiplication.

The main contributions of this work are summarized as follows. 1) In terms of computing accuracy, an optimized coding sequence is developed to work with the thermometer code. 2) A serial multiplier and parallel multiplier are

This work was supported by the Fundamental Research Funds for the Central Universities of China (Grant No. JZ2020HGTA0085, No. JZ2020HGQA0162), and by the Natural Sciences and Engineering Research Council (NSERC) of Canada (Project Number: RES0048688).

Y. Zhang, L. Xie, and G. Xie are with the School of Microelectronics, Hefei University of Technology, Hefei 230009, China (e-mail: ahzhangyq@hfut.edu.cn; 2098425338@qq.com; gjxie8005@hfut.edu.cn)

J. Han is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 1H9, Canada (e-mail: jhan8@ualberta.ca)

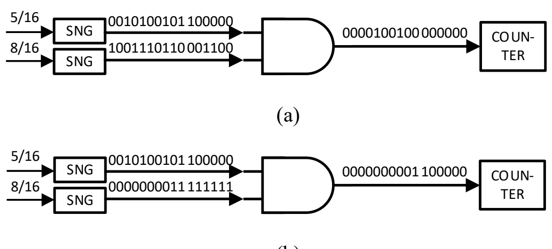


Figure 1. (a) A classical stochastic multiplier, (b) performing multiplication using thermometer codes.

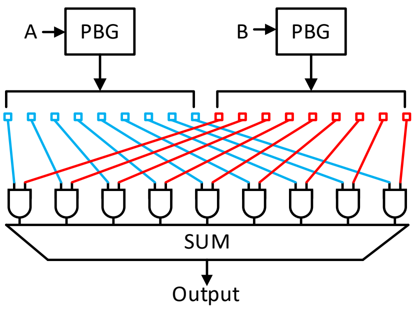


Figure 2. The parallel multiplier in [7].

proposed to, respectively, reduce area and latency. The serial multiplier inherits the advantages of Sim's multiplier and the parallel multiplier can obtain results in one clock cycle, which greatly reduces computation time and improves energy efficiency. 3) These two multipliers are applied to image multiplication to show their practicability in real applications.

This paper proceeds as follows. Section II describes the optimized encoding sequence, and the proposed serial and parallel multipliers. Section III shows the experimental results and an application of various multipliers. Section IV concludes this paper.

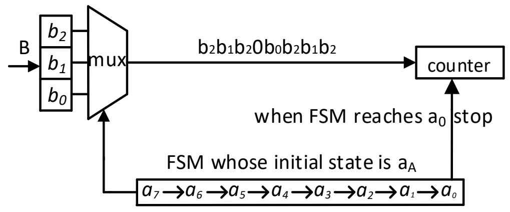
II. The Proposed Multipliers

A. Optimized Encoding Sequence

Take the 3-bit binary input $b_2b_1b_0$ as an example. We take the low discrepancy coding sequence $b_2b_1b_2b_0b_2b_1b_20$ in [6] as the initial sequence, and then perform the full permutation of the initial sequence to obtain a total of $C_8^4 \times C_4^2 \times C_2^1 = 840$ permutation cases. Next, the MSE of each permutation (including the initial sequence) and the thermometer code after performing multiplication is calculated. Record all the permutations that are smaller than the MSE of the initial sequence, among which the permutation with the smallest MSE is our optimized coding sequence.

B. The Proposed Serial Multiplier

Fig. 3 shows the circuit of the proposed 3-bit serial multiplier. B generates the optimized coding sequence $b_2b_1b_20b_0b_2b_1b_2$ serially by an FSM and MUX of length 2^n. Another input A in this FSM is the initial state a_A (for example, if A=3, the initial state is a_3). In each clock cycle, the state machine jumps to the next state while determining the sequence value generated under the current clock cycle: adding 1 to the output counter if it is 1; otherwise keeping it unchanged. When the state machine reaches a_0, the output counter produces the desired multiplication result. This multiplier requires only one FSM to control both bitstream generation and output generation. It also has the advantage of being able to end early. Compared to Sim’s design, it no longer requires an additional down-counter, thus further reducing the area, while improving its computing accuracy by using the optimized encoding sequence.


B
b_2
b_1
b_0
mux
b₂b₁b₂0b₀b₂b₁b₂
counter
when FSM reaches a_0 stop
FSM whose initial state is a_A
$a_7 \to a_6 \to a_5 \to a_4 \to a_3 \to a_2 \to a_1 \to a_0$
Figure 3. The proposed serial multiplier.

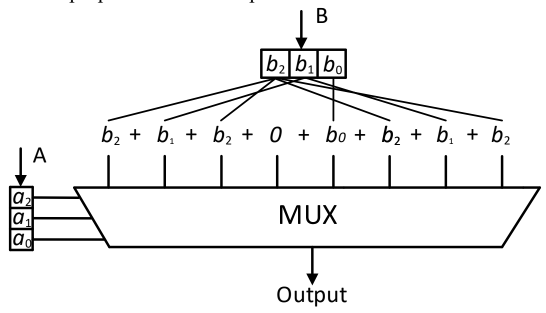


Figure 4. The proposed parallel multiplier.

TABLE I. The MSE of Various Lengths of the Optimized and Sim Sequences

n-bit / MSE	3 (10^{-3})	4 (10^{-4})	5 (10^{-4})	6 (10^{-5})	7 (10^{-5})	8 (10^{-6})
Sim	2.30	6.75	1.92	5.37	1.47	4.02
Proposed	1.50	4.61	1.35	3.89	1.10	3.07

C. The Proposed Parallel Multiplier

High latency is a drawback of serial stochastic circuits. In order to further reduce the latency, we consider the use of parallel stochastic bitstreams. To this end, a parallel multiplier is proposed, as shown in Fig. 4. Take 3-bit streams as an example. It uses wire connections to directly generate the optimized coding sequence $b_2b_1b_20b_0b_2b_1b_2$ for the binary input B, which saves the additional PBGs required to generate the bitstreams. The sum of the sequences is then computed in one clock cycle. Note that the sum of the sequence here includes the sum of all the previous parts. In other words, the computed sum of the first 8 bits includes the sum of the first 2 to 7 bits. In this way, the other binary input A can be directly used as the select signal for the MUX.

III. Simulation Results and Application

A. Computing Accuracy

In order to compare the computing accuracy of the proposed optimized encoding sequences with the low discrepancy sequences of Sim, the MSE is measured for respectively multiplying them with the thermometer codes [9]. The thermometer codes are chosen to perform the multiplication with them because the characteristics of thermometer codes allow the multiplication to end earlier in hardware. The MSE is given by

$$MSE = \sum_{x=0}^{2^n-1}\sum_{y=0}^{2^n-1}\frac{(P_{xy}-P_x\cdot P_y)^2}{2^n\times 2^n}, \quad (1)$$

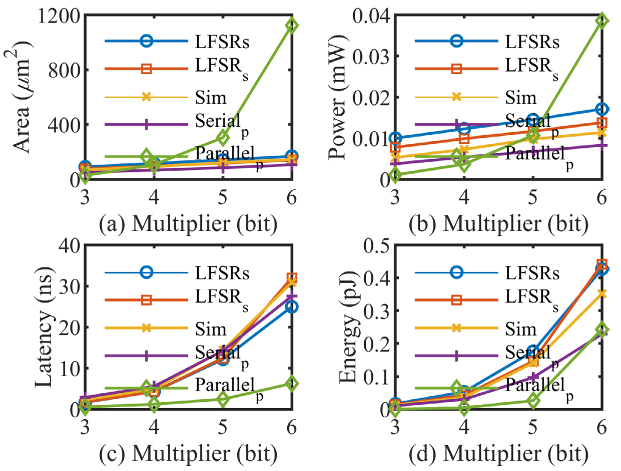


Figure 5. Performance of various multipliers with different bit-lengths. (a) Area. (b) Power. (c) Latency. (d) Energy.

where P_x and P_y represent the decimal value of the optimized encoding sequences (or Sim's encoding sequences) and the thermometer codes, respectively, and P_{xy} represents the value by ANDing two types of encoding sequences.

Experimental results show that when the input binary number is $b_2b_1b_0$, the minimum value of MSE in its fully aligned bitstream is 0.0015. The corresponding optimized coding sequences are $b_2b_1b_20b_0b_2b_1b_2$ and $b_2b_1b_2b_00b_2b_1b_2$, while the MSE of Sim's bitstream alignment $b_2b_1b_2b_0b_2b_1b_20$ is 0.0023, which is inferior to the proposed optimized coding sequences in terms of accuracy. Comparing the optimized coding sequences with Sim's sequence, it can be found that the position of 0 in the sequence is more accurate in the middle of the bitstream than at the end of the bitstream. For example, when calculating (5/8)×(5/8), Sim's coded bitstream 10111010 and the thermometer code 111111000 are ANDed to give a result of 4/8, while the optimized coded bitstream 10110101 or 10101101 and the thermometer code 111111000 are ANDed to give a result of 3/8. It is obvious that 3/8 is closer to the exact value of 3.125/8. TABLE I gives the MSE of the optimized encoding sequence and Sim's encoding for 3- to 8-bit sequences. As can be seen, although the encoding sequences of Sim already produce a small error in the multiplication, it is clear that the optimized encoding sequences perform the multiplication with a lower error. The MSE is reduced by 29% on average for 3- to 8-bit multipliers.

B. *Hardware Cost*

The hardware of all multipliers is measured by the Synopsys Design Compiler with TSMC's 40 nm standard library. These multipliers use a binary input-output interface including two LFSRs, a shared LFSR (LFSR_s) [4], Sim [6], the proposed serial (Serial_p), and parallel multipliers (Parallel_p). They are directly applied to image multiplication. The results show that the proposed multiplier obtains an approximate image at 6 bits as well as a very close approximation to the exact image, so we design only up to 6 bits in the hardware comparison. Fig. 5 shows the performance of various multipliers in terms of area, power, latency, and energy consumption as the number of input bits increases. It can be seen that the increase in the number of input bits exponentially increases the length of the bitstream, resulting in an increasing trend in the area, power, latency, and energy consumption for all multipliers. However, the proposed

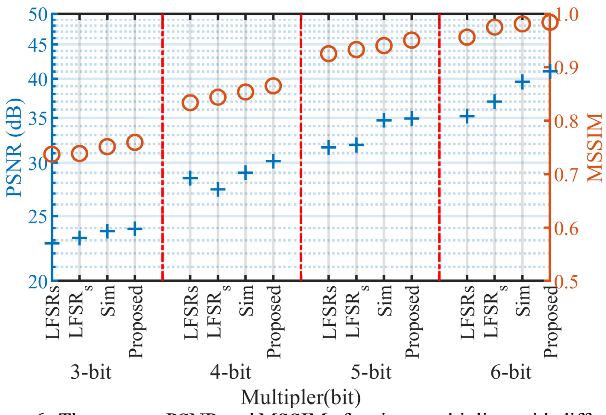


Figure 6. The average PSNR and MSSIM of various multipliers with different bit-lengths to implement image multiplication.

parallel multipliers have the lowest latency and energy; the proposed serial multipliers have the lowest area and power. Compared to Sim's multiplier, the proposed parallel multiplier reduces latency and energy by 78% and 57% on average, respectively; the proposed serial multiplier reduces area and energy by 24% and 28% on average, respectively. Therefore, we recommend the proposed serial multiplier if low area and low energy are pursued, and the proposed parallel multiplier if low latency and low energy consumption are pursued.

C. *Image Multiplication*

To show the efficacy of the proposed multipliers in practical applications, image multiplication is considered. To evaluate the quality of output images, the peak signal-to-noise ratio (PSNR) and mean structural similarity index metric (MSSIM) are used here [10]. The PSNR is given by

$$PSNR = 10\log_{10}\left(\frac{w\cdot r\cdot MAX^2}{\sum_{i=0}^{w-1}\sum_{j=0}^{r-1}\left[S'(i,j)-S(i,j)\right]^2}\right), \tag{2}$$

where w and r denote the image dimensions, $S'(i,j)$ and $S(i,j)$ are respectively the exactly and stochastically computed values of output pixels, and MAX is the maximum value of output pixels. The MSSIM is defined as

$$MSSIM(X,Y) = \frac{1}{k}\sum_{i=1}^{k}\frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{3}$$

where x and y represent the windows of the exact and approximate images, respectively. The larger the PSNR and MSSIM, the closer the output image is to the exact image.

To reduce the fluctuation in data, a combination of 5 images is tested for image multiplication. The pixel values of the same position in two images are used as inputs to the multiplier. Fig. 6 shows the average of the PSNR and MMSIM of the image multiplications obtained using 3 to 6-bit SC multipliers (PSNR marked with '+' and MSSIM marked with 'O'). Both the proposed serial and parallel multipliers use the optimized sequences to implement the multiplication, so they do not lead to different image quality and are both denoted as the "proposed" in Fig. 6. In addition, the method in [9] that minimizes the SC correlation (SCC) is used here to share an LFSR [11]. It can be seen from Fig. 6 that both MSSIM and

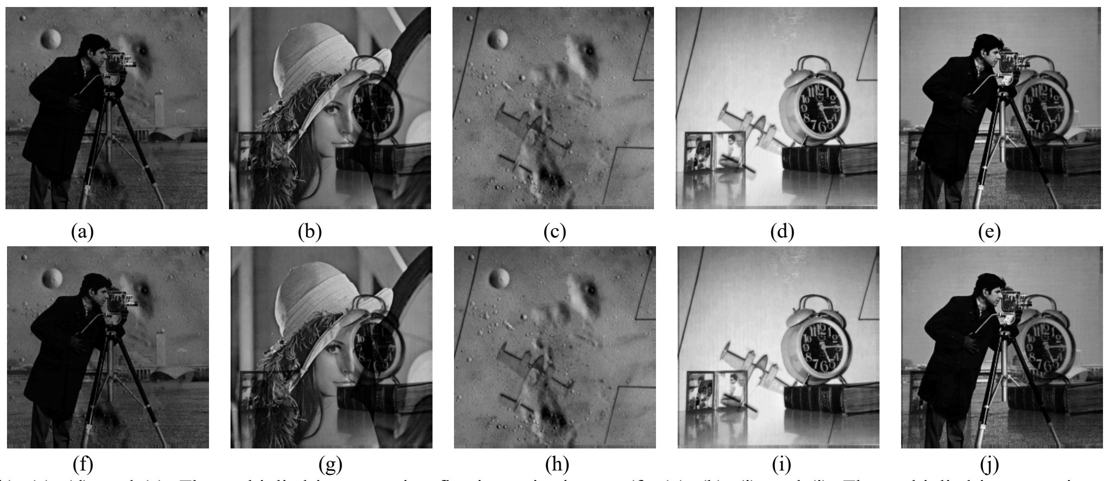


Figure 7. (a), (b), (c), (d), and (e): The multiplied images using floating-point inputs. (f), (g), (h), (i), and (j): The multiplied images using the proposed multiplier.

PSNR show an increasing trend as the number of input bits increases. The proposed design results in the highest PSNR and MSSIM for the same input bit-lengths. In addition, when the input of the multipliers reaches 6 bits, the proposed design yields an image with an average PSNR of more than 40 and an average MSSIM of more than 0.98, which cannot be achieved by the other multipliers. This means that the proposed multiplier produces higher quality images compared to other multipliers. Fig. 7 shows the output images obtained by the multipliers with 6-bit inputs when performing image multiplication (Cameraman × Moon, Lena × Clock, Airplane × Moon, Airplane × Clock, Cameraman × Clock) downloaded from The USC-SIPI Image Database [12]. Fig. 7 (a) to (e) are the exact images obtained from floating-point multipliers, while (f) to (j) are the approximate images obtained by using the proposed multipliers. As can be seen in Fig. 7, in contrast to exact image multiplication, only the proposed 6-bit multiplier is required to obtain results that are not significantly different from the exact image.

IV. Conclusion

In this paper, optimized encoding sequences with higher computing accuracy are introduced to more efficiently implement stochastic multiplication, making the optimized serial design with lower area and power without adding other hardware overheads. In addition, a new parallel multiplier is proposed so that the computation results are obtained in one clock cycle, which greatly reduces the circuit delay and thus the circuit energy consumption. Both multipliers use the optimized encoding sequences to guarantee a high computing accuracy. Experimental results show that compared to Sim's multiplier, the latency and energy consumption of the proposed parallel multiplier are reduced by 78% and 57% on average, respectively; the area and energy consumption of the proposed serial multiplier are reduced by 28% and 24% on average, respectively. At the same time, the computing accuracy is improved by 29% on average. Finally, the multipliers are applied to image multiplication, and the results show that the proposed multipliers produce output images with higher quality, compared to existing designs. Therefore, we suggest the proposed serial multiplier when low area and low energy consumption are desired, and the proposed parallel multiplier when low latency and low energy consumption are desired.

References

[1] 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.

[2] Y. Liu, L. Liu, F. Lombardi, and J. Han, “An energy-efficient and noise-tolerant recurrent neural network using stochastic computing,” *IEEE Trans. Very Large Scale Integr. VLSI Syst.,* vol. 27, no. 9, pp. 2213-2221, Sep. 2019.

[3] P. Schober, M. Najafi, and N. Taherinejad, “High-accuracy multiply-accumulate (MAC) technique for unary stochastic computing,” *IEEE Trans. Comput.*, pp. 1-1, Jun. 2021.

[4] H. Ichihara, S. Ishii, D. Sunamori, T. Iwagaki, and T. Inoue, “Compact and accurate stochastic circuits with shared random number sources,” in 2014 IEEE 32nd International Conference on Computer Design (ICCD), Seoul, South Korea, 2014, pp. 361-366.

[5] S. Hyeonuk, and L. Jongeun, “A new stochastic computing multiplier with application to deep convolutional neural networks,” in the 2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC), Austin, TX, USA, 2017, pp. 29-34.

[6] H. Sim, and J. Lee, “Cost-effective stochastic MAC circuits for deep neural networks,” *Neural Netw.,* vol. 117, pp. 152-162, Sep. 2019.

[7] Y. Zhang, R. Wang, X. Zhang, Y. Wang, and R. Huang, “Parallel hybrid stochastic-binary-based neural network accelerators,” *IEEE Trans. Circuits Syst. II Express Briefs,* vol. 67, no. 12, pp. 3387-3391, Dec. 2020.

[8] M. Najafi, D. Jenson, D. Lilja, and M. Riedel, “Performing stochastic computation deterministically,” *IEEE Trans. Very Large Scale Integr. VLSI Syst.,* vol. 27, no. 12, pp. 2925-2938, Dec. 2019.

[9] S. Salehi, “Low-cost stochastic number generators for stochastic computing,” *IEEE Trans. Very Large Scale Integr. VLSI Syst.,* vol. 28, no. 4, pp. 992-1001, Apr. 2020.

[10] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Trans. Image Process.,* vol. 13, no. 4, pp. 600-12, Apr 2004.

[11] 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.

[12] "The usc-sipi image database," https://sipi.usc.edu/database/.