Approximate Divider Design Based on Counting-Based Stochastic Computing Division

Shuyuan Yu*, Yibo Liu*, Sheldon X.-D. Tan*
* Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521 stan@ece.ucr.edu

2021 ACM/IEEE 3rd Workshop on Machine Learning for CAD (MLCAD) | 978-1-6654-3166-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/MLCAD52597.2021.9531079

***Abstract*—Stochastic computing (SC) promises extremely low cost and energy efficiency for error-tolerant arithmetic operations in many emerging applications such as image processing and deep neural networks. Existing SC-based nonlinear functions like division, however, require highly correlated bit-streams, which does not fit well with the existing SC computing framework in which randomness is required for accuracy. In this paper, we propose a novel SC-based divider design based on recently proposed counting-based stochastic computing scheme, which is much more accurate and faster than traditional SC, and does not depend on randomness of bit-streams for accuracy. We show how such counting-based SC can be applied to nonlinear functions like division. The new divider, called counting-based divider, or *CBDIV*, exploits both the correlation requirement of existing SC-based division methods and high efficiency of counting-based SC scheme. It essentially combines the best of two worlds in SC and the resulting division operation can be performed as a more efficient partial counting process. Experimental results show that the proposed CBDIV implemented in a 32nm technology node outperforms state of art works by 77.8% in accuracy, 37.1% in delay, 21.5% in area, 50.6% in ADP (area delay product) and 25.9% in power. CBDIV also saves 31.9% in energy consumption when compared to the fixed-point division baseline, and is much more energy efficient than existing SC-based dividers for binary inputs and outputs required in efficient image process implementations. Furthermore, CBDIV with 5-bit precision can even outperform state of art works with 7-bit precision in accuracy by 15.4%. Finally, we compare CBDIV with other state of art SC dividers in contrast stretch application and show that CBDIV can improve the accuracy with 20.6dB in average, which is a huge improvement.**

I. Introduction

Stochastic computing (SC) is a promising low-cost, error-resilient alternative to the conventional binary based computing. Compared to binary design, SC is shown to have better error resilience, progressive trade-off among performance, accuracy and energy, as well as cheap implementation of complex arithmetic operations. Fig. 1 shows the conventional stochastic computing (SC) multiplier, where the number or stochastic number (SN), is represented by a bit-stream, whose signal probability, or frequency of bit '1', determines its value. Naturally, the value is defined in the range $[0, 1]$, called unipolar, or over $[-1, 1]$, called bipolar. For instance, in Fig. 1, the number X represents $4/8$ as we have four bit '1's in the bit-stream of 8 bits. One of the major benefits for SC is that many arithmetic operations such as multiplication can be simply implemented by *AND* operation (or *XNOR* gate for bipolar) as shown in this figure. SC has been applied to error-correcting codes [1], image processing [2], and recently deep neural networks (DNNs) [3]–[6].

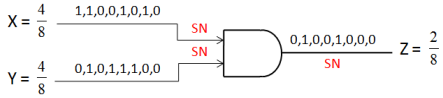

Fig. 1: Conventional SC multiplication.

However, SC's simple hardware implementation suffers from several drawbacks: first, traditional SC takes 2^N cycles to deal with N-bit precision inputs (as shown in Fig. 1), which can be much larger than binary logic especially when N is large. So SC is still more suitable for applications which don't require high accuracy. Second, the accuracy of SC process requires the randomness of two bit-streams (ideally with zero correlation) in addition to the length of bit-stream. As a result, many research works have been proposed to develop high-quality random number generators (RNGs) that exhibit zero or close to zero correlation, including low-discrepancy sequences [7], [8], bit scrambling methods [9], [10].

For SC-based division operations, recent studies showed that they could be performed much more efficiently when the two input bit-streams were highly correlated so the division could be approximated by computing the conditional probability of the two streams [11], [12]. To mitigate the different correlation requirements between traditional SC division, a translation module (called skewed synchronizer) was introduced to resolve this problem at the cost of higher area overhead [12].

Recently, a more efficient and also accurate SC multiplier was proposed to partially mitigate the two aforementioned problems in traditional SC [5]. Instead of using an *AND* gate for the multiplication of two bit-streams, the new multiplier essentially counts bit '1's in one bit-stream based on the value of the other bit-stream. Further more, the bit-stream to be counted can be generated in a deterministic way based on a finite state machine (FSM). As a result, the whole design is simplified into two counters and a simple bit-stream generator. In this work, we call this design *counting-based SC multiplier* (CBSC-Multiplier). CBSC-Multiplier brings two important benefits: first, it does not require the randomness of the two bit-streams anymore without loss of accuracy, which turns out to be very compatible with the correlation requirement of existing SC-based methods. Second, it can be even faster than traditional SC as it can perform early

This work is supported in part by NSF grants under No. CCF-1816361, in part by NSF grant under No. CCF-2007135 and No. OISE-1854276.

termination (partial counting) by design, which can't be easily achieved in the traditional SC methods.

Based on this observation, we propose a novel SC-based division design, called counting-based divider, *CBDIV*, which exploits both the correlation requirement of existing SC-based division methods and the high efficiency of the counting-based SC scheme. In other words, we get the best of two worlds in SC and the resulting division operation can be performed as a much more efficient partial counting process. As a result, we extend the CBSC design to an important nonlinear arithmetic operation for more general counting-based stochastic computing. The proposed CBDIV design works seamlessly with CBSC multiplication without the translation required by the existing methods. It also fits well with hybrid computing in which binary and SC bit-streams are both required to leverage different arithmetic operations for best overall efficiency. For instance, multiplication and nonlinear functions are more efficient in SC, while addition is more efficient in binary form. The key contributions of this work are as follows:

1. First, we develop a new counting-based SC division design, *CBDIV*, which is based on the recently proposed correlated division design [11] and the CBSC concept. The new design can be performed in a counting process, with reduced operation time and improved accuracy.
2. Second, experimental results show that the proposed CBDIV implemented in a 32nm technology node outperforms state of art SC dividers by 37.1% in delay, 21.5% in area, 50.6% in ADP and 25.9% in power. CBDIV also saves 31.9% in energy consumption when compared to the fixed-point division design.
3. Third, we use RMSE (root mean square error) to evaluate the error behavior of CBDIV and compare it with state of art works and the fixed-point divider. Numerical results show that CBDIV outperforms state of art works by at least 77.8% in average. And CBDIV with 5-bit precision can even outperform state of art works with 7-bit precision in accuracy by 15.4%.
4. Forth, we evaluate the CBDIV method with contrast stretch application, and compare the proposed method with state of art works. Experimental results show that CBDIV can improve accuracy by 20.6dB in average when compared to previous works.

This paper is organized as follows: Section II reviews the CBSC concept and recently proposed SC division methods. Section III presents the proposed CBDIV design including techniques to further save area. Experimental results for the hardware performance, error metrics of CBDIV design and the comparison with the binary logic baseline, state of art works are summarized in Section IV. Section V concludes this paper.

II. Review of Related Works

In this section, we review a few related works concerning our proposed new division design.

A. Counting-based SC multiplication

Assume the precision is n-bit for the given two binary numbers (BNs) x and w, which are both in the range of [0, 1]. The conventional SC multiplier using *AND* gate (for unipolar encoding) will take 2^n, which is the length of bit-stream of stochastic number (SN), cycles to finish the work. To improve this, Sim *et al.* in [5] proposed a counting-based SC (CBSC) multiplier design with high accuracy. The multiplier consists of two counters and an FSM based stochastic number generator (SNG), shown in Fig. 2(c). The SNG is used to generate a low-discrepancy SN bit-stream of x. The authors proposed a deterministic way to do this. The method evenly distributes the x_{i-1}, which is the ith bit of x, based on its binary weight 2^{i-1}. For instance, if $i = 3$, then x_2 will appear 4 times in the resulting SN as shown in Fig. 2(a). Such SN generation can be simplified and implemented by an FSM and a MUX. The FSM is actually an up counter counts from 0 to $2^n - 1$. The MUX then outputs x_{i-1} based on the output value of the FSM.

If the SN bit-stream for w is set to be series of '1' followed by a bunch of '0' as shown in Fig. 2(b). As SC multiplication is simply *AND* operation, it is no necessary to count the second half of the output bit-stream. The first half of the operation takes $w \cdot 2^n$ cycles. So, the whole counting process only requires the same latency to finish, which saves a lot of computing time. The authors used a down counter to realize the idea. While $w \cdot 2^n$ is used as the initial value, the down counter decreases by one in each clock cycle. When it reaches "zero", the process is terminated. As a result, such design is simpler as one traditional SNG (typically using Linear Feedback Shift Registers) and *AND* gates are eliminated in exchange of a down counter, which is much cheaper than SNG.

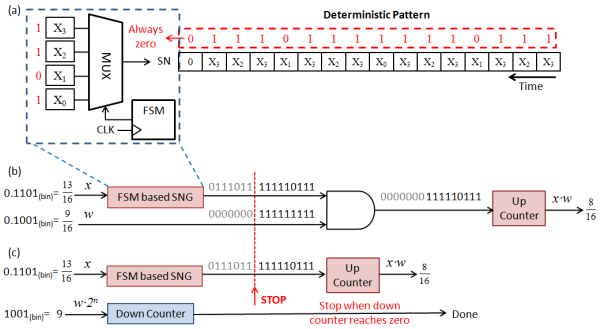

Fig. 2: (a) FSM based SNG which generates SN bit-stream in a deterministic way. (b) CBSC concept. (c) CBSC multiplication method.

B. SC-based division

Traditional SC division follows Gaines' design [13], in which an up-down counter is used to reach an equilibrium when two SN numbers, $x_{1,SN}$ and $x_{2,SN} \cdot p_{z,SN}$ are equal. Then $p_{z,SN} = x_{1,SN}/x_{2,SN}$. So it needs SC multiplication for $x_{2,SN} \cdot p_{z,SN}$. Gaines' design was proved to take long time to converge [11] and required both the dividend and the divisor to be uncorrelated. A recent work proposed by Temenos [14] mitigated these issues by using a deterministic FSM-based stochastic division design (DFSM-DIV), which did not require the independence between the input SN bit-streams. However,

DFSM-DIV still required at least 3 orders of magnitude (10^2) clock cycles to converge in average and could not achieve much accuracy improvement.

Another idea, which is correlation based, made improvement in a different way. *Correlated division*, or CORDIV, proposed by Te-Hsuan [11] explored the fact that division could be computed as the conditional probability of two SN numbers: $P_{X_1|X_2}$. The formula can be simplified to

$$P_{X_1|X_2} = p_{x_1,x_2}/p_{x_2} = x_{1,SN}/x_{2,SN} \quad (1)$$

when x_1 and x_2 are highly correlated, where p_{x_2} is the event of x_2 and p_{x_1,x_2} is the joint event of x_1 and x_2. As a result, the output probability $P_{Quotient}$ can be expressed by the ratio of number of bit '1' in the dividend and the divisor SN bit-streams: $N^1_{Dividend}/N^1_{Divisor}$. Thus, CORDIV requires that the divisor is always larger than the dividend to avoid the quotient being larger than one. This work proved that with highly correlated input SN bit-streams, CORDIV could obtain much more accurate results when compared with Gaines' design [13]. But the accuracy of CORDIV method depends on the quality of input SN bit-streams.

Specially designed SNGs shown in Fig. 3 [11] has been proved that could mitigate this problem. However, the accuracy still depends on the distribution of bit '1' in input SN bit-streams, shown in Fig. 4. Adding skewed synchronizer could improve accuracy, but was still not accurate enough, especially in certain output value ranges [12].

Besides, all these SC-based division designs suffered from accuracy varying among different output value ranges and the problem of large latency, taking 2^N cycles to finish the process with inputs whose corresponding binary precision are N-bit, which is an intrinsic shortcoming of SC implementation. Furthermore, the energy consumption is also high when the inputs and outputs are in binary form to interface with other computing parts in typical image process application [15].

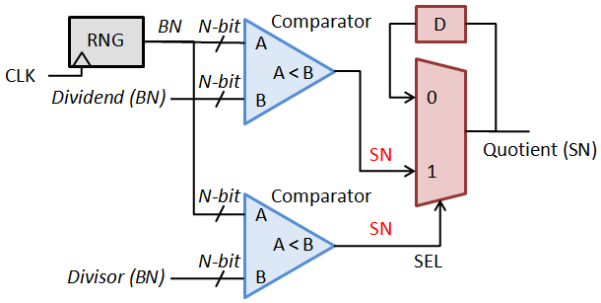

Fig. 3: CORDIV design diagram.

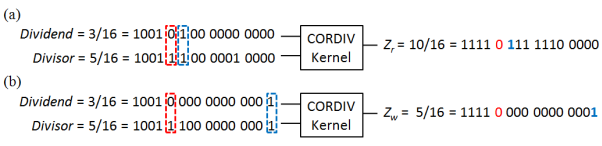

Fig. 4: CORDIV method input SN bit-streams with different bit distribution: (a) Quotient result with appropriate bit distribution. (b) Quotient result with extreme bit distribution.

III. Proposed Counting-Based SC Division Design

In this section, we try to mitigate the aforementioned issues in the existing correlated division designs and propose a new SC division method based on a more efficient counting-based SC (CBSC) concept.

From Fig. 4, we can see that the location of bit '1' in the divisor may significantly affect the quotient accuracy. The two input probabilities are 3/16 and 5/16, respectively, whose exact quotient result would be 0.6. The first 1-0 pair and the following 1-1 pair in the divisor and the dividend SN bit-streams are in red and blue blocks, respectively. And the output in the quotient bit-stream at the corresponding location, which are the first bit '0' and following '1' are also marked with red and blue, respectively. As CORDIV method will consistently output bit '0' if the divisor and the dividend SN have 1-0 pair followed by 0-0 pairs at the corresponding location until a 1-1 pair appears. And will consistently output '1' if the two input bit-streams have a 1-1 pair followed by 0-0 pairs until a 1-0 pair appears. Suppose the input SNs have appropriate bit '1' distribution as shown in the upper case in Fig. 4, CORDIV computes Z_r = 10/16 = 0.625. But if we have extreme bit '1' distribution given as the lower case in Fig. 4, too many 0-0 pairs appear between 1-0 pair and the following 1-1 pair, CORDIV computes Z_w = 5/16 = 0.3125, which is far from the exact result.

To solve this problem, we propose a new division method called CBDIV, which stands for *Counting-Based Division*. Unlike CORDIV [11] or ISCBDIV [12], we don't require a synchronizer or expensive RNG-comparator structure to guarantee a strong correlation of two input SN bit-streams. Instead, the SN generation in CBDIV utilizes a deterministic pattern shown in Fig. 2(a), which is isolated from the random fluctuations.

Since we use the deterministic pattern to generate the divisor SN bit-stream, the location of bit '1' is fixed. We can easily arrange the dividend SN bit-stream to favor the 1-1, 0-0 pairs as many and fast as possible, as the location of bit '1' in the dividend SN bit-stream is completely determined according to the divisor SN. That means if a bit in divisor SN is '1', the bit in dividend SN at the corresponding location will also be set to '1' to guarantee all the 1-1 pairs to be continuously found. So, before $S_{Dividend}$ has used up all bit '1's in the SN bit-stream, $S_{Dividend}$ is completely the same as $S_{Divisor}$. In other words, when the first 1-0 pair occurs, meaning that the dividend (which is less than divisor by design) has already used up all the bit '1' and no more 1-1 pair is left, which can be illustrated in an example shown in the first 2 rows in Fig. 5(a). Here, the value of the dividend SN $S_{Dividend}$ and the divisor SN $S_{Divisor}$ are 6/16 and 9/16, respectively.

As we can see, after the arrangement, the right parts of $S_{Divisor}$ and $S_{Dividend}$ in the blue dash line block in Fig. 5(a) are exactly the same. Then based on CORDIV method [11], the quotient SN bit-stream $S_{Quotient}$ simply obtains all its bit '1's in the right half at the 3rd row in Fig. 5(a). A counter which increases by 1 in every clock cycle is enough to get the binary form value, which is shown in the 4th row in Fig. 5(a).

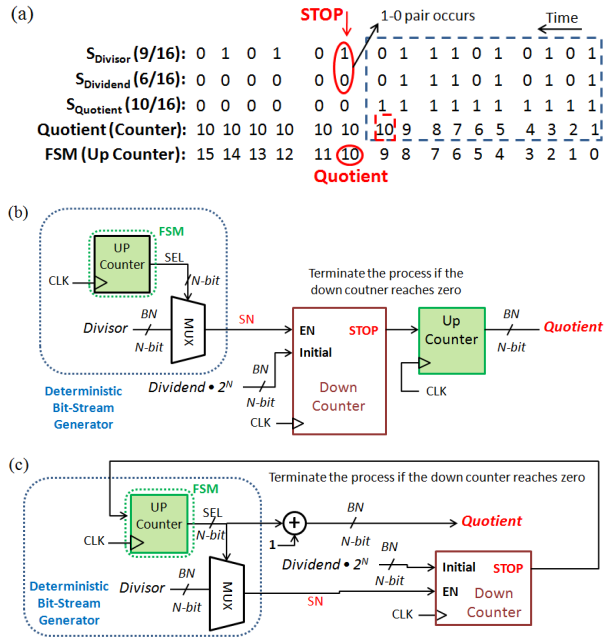

Fig. 5: (a) The proposed CBDIV method. (b) The proposed CBDIV design. (c) The optimized CBDIV design.

We notice that the remaining bits outside the blue frame in $S_{Quotient}$ are all bit '0'. As a result, we don't need to count anymore and can simply terminate the process, thus reduce the computing time. The result now counts 10/16 = 0.625, which is a good approximation to the exact result 2/3 = 0.667.

Based on our CBDIV method, the CBDIV design actually requires 3 counters to realize the process, 2 up counters and 1 down counter, as shown in Fig. 5(b). One up counter is used as an FSM (finite state machine) to generate the divisor SN bit-stream. The bit-stream follows similar low discrepancy distribution proposed in [5]. Because the two input bit-streams, $S_{Dividend}$ and $S_{Divisor}$, are completely the same before the division process ends as we have discussed before. We only need to generate one of them. Here, we only generate $S_{Divisor}$. The other up counter is used to convert the quotient SN bit-stream $S_{Quotient}$ to binary form. It simply increases by one in every clock cycle since the bits in $S_{Quotient}$ are always '1' before the division process ends. The down counter, with an initial value of $Dividend \cdot 2^N$, determines when to terminate the process. The enable signal "EN" of the down counter is connected to the $S_{Divisor}$ signal. So, the down counter will decrease by one if bit '1' appears in $S_{Divisor}$.

By observing the 4th and 5th row in Fig. 5(a), we notice that the FSM output is always one smaller than the output quotient value before the process terminates, which means we can infer the quotient from the FSM output value even without a counter. So we combine the 2 up counters together. Considering the $S_{Divisor}$ generation, we keep the up counter used as the FSM. When the division process finishes, we simply add one to the output of the FSM to obtain the quotient. Now we only require 2 counters to form our CBDIV design. The optimized CBDIV design is shown in Fig. 5(c).

IV. Experimental Results and Discussions

In this section, we evaluate the performance of the proposed *counting-based SC divider*, CBDIV. We also compare CBDIV against the binary logic baseline and the state-of-the-art works.

A. Experimental setup

To evaluate the performance of the proposed CBDIV design, we compare the error behavior and the hardware performance of CBDIV with a fixed-point divider using long division algorithm. The area, the critical path, the average latency (clock cycles), the average delay, the area and delay product (ADP), the power and the total energy consumption are measured to evaluate the hardware performance of CBDIV. We also compare CBDIV with several state of art works such as DFSM-DIV (*SC division with deterministic finite state machines*) [14], CORDIV [11], and ISCBDIV [12]. All the approaches are with binary number inputs of 7-bit precision (corresponding to 128-bit length bit-stream).

The dividers are implemented in Verilog and synthesized with Synopsys Design Compiler using EDK 32nm standard cell library [16]. For fair comparison, all the dividers are binary in and binary out, SNGs which generate SN bit-streams and a counter which counts the value of the result SN bit-stream have been added to the in-stream SC-based division designs. The hardware performance comparison of the dividers are shown in Table I. For fair power and energy consumption comparison, the input clock frequency of all the dividers are set to be 100MHz. In addition, energy consumption is measured based on the total execution time and its power consumed during the whole division process.

The error behavior is measured using *root mean square error* (RMSE). To evaluate the error behavior, we develop behavioral simulation models for all the SC-based dividers listed in Table I in MATLAB and measure the accuracy of over 100, 000 operations. To promise the dividend be smaller than the divisor, we generate the dividend in the range of [0, 0.5], and the divisor in the range of [0.5, 1].

B. Hardware performance

From the second column in Table I, we observe that among all the listed stochastic dividers, our proposed CBDIV is the most area efficient, doing 72.6% of area reduction when compared to the fixed-point divider (binary logic) baseline, and outperforms state of art stochastic dividers by at least 21.5%. As the latency of CBDIV design is determined by the value of the dividend, which is proved in Sec. III, the latency of CBDIV design will vary according to the inputs. This is different from the latency of previous SC division designs when certain precision is determined, which is a fixed value. We show this property in Fig. 6. So we use $Avg.\ Latency$ (average latency) and $Avg.\ Delay$ to measure the average computing time of CBDIV and compare these two values against previous works. From column 4, we can see that CBDIV saves 65.6% of latency in average; and from column 5, the $Avg.\ Delay$ of CBDIV is 75.68ns, outperforms state of art works by at least 37.1%. Also, though the latency of

	Area (μm^2)	Critical Path (ns)	Avg. Latency (cycles)	Avg. Delay (ns)	ADP	Power (μW)	Avg. Energy (pJ)
Fixed-Point	1090	3.16	17	53.72	58555	41.4	7.038
DFSM-DIV	520	1.24	128	158.72	82534	22.3	28.544
CORDIV	381	0.94	128	120.32	45842	14.7	18.765
ISCBDIV	394	0.98	128	125.44	49423	15.8	20.211
CBDIV (proposed)	**299**	**1.72**	**44**	**75.68**	**22628**	**10.9**	**4.796**

TABLE I: Hardware performance for SC and fixed-point dividers.

CBDIV is 40.9% larger than which of the fixed-point divider, the area and delay product (ADP), is much smaller than the fixed-point divider (61.4% smaller), and improves by at least 50.6% when compared to previous works as shown in column 6.

From the last two columns in Table I, we observe that CB-DIV performs the best both in power and energy consumption among all the listed division designs. In power aspect, CBDIV outperforms state of art works by 25.9%. For the energy consumption aspect, we note that in-stream SC-based division designs will cost even more energy than the fixed-point divider when SNGs and the counter are taken into consideration. CBDIV design has solved this problem by improving the computing latency, doing 31.9% in energy reduction when compared to the fixed-point divider baseline.

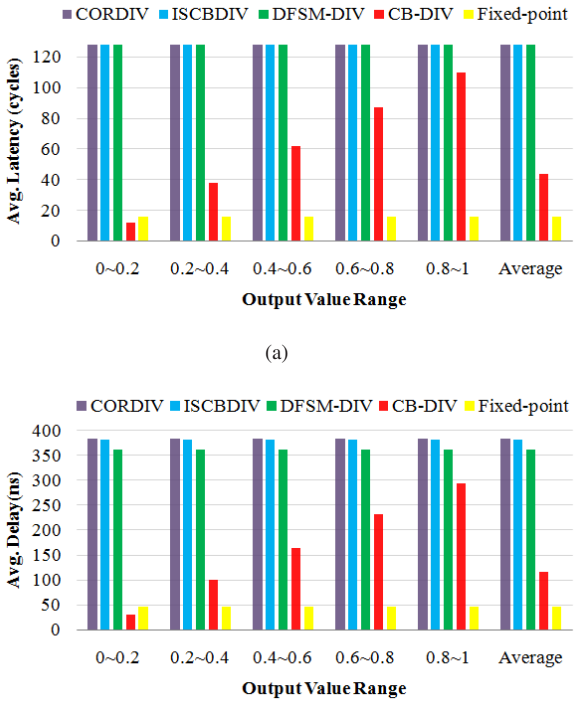

Fig. 6: Latency and delay comparison among CBDIV and other dividers with different output value ranges (7-bit precision): (a) Average latency. (b) Average delay of the whole process.

C. Error behavior

Fig. 7 shows the accuracy comparison among the proposed CBDIV method with the baseline fixed-point divider and existing approaches: CORDIV [11], ISCBDIV [12], DFSM-DIV [14]. LFSR (linear feedback shift register) is utilized to generate SN bit-streams for binary inputs in state of art approaches for comparison.

We show the results over five non-overlapped output value ranges as well as the average RMSE value obtained from 100, 000 computations for all the algorithms. From Fig. 7, we can observe that CBDIV outperforms state of art works in all of the five non-overlapped output value ranges. Compared with DFSM-DIV, which has the smallest RMSE value in average, CBDIV makes an improvement by 77.8%. To make a supplement, note in Fig. 7, CBDIV shows similar RMSE values in different output value ranges, different from performances of the existing works which is due to the intrinsic property of different SC division methods.

Furthermore, we show the error behavior of CBDIV with different bit-stream length (precision) compared with the fixed-point baseline and state of art works in Fig. 8. In Fig. 8, we show that even with 5-bit precision, the average value of RMSE of CBDIV is still 15.4% better than DFSM-DIV with 7-bit precision.

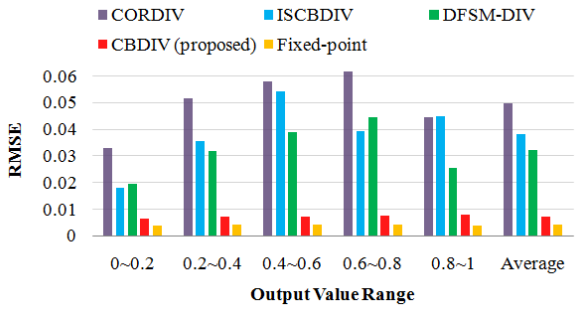

Fig. 7: Error behavior of CBDIV and other dividers with different output value ranges (7-bit precision).

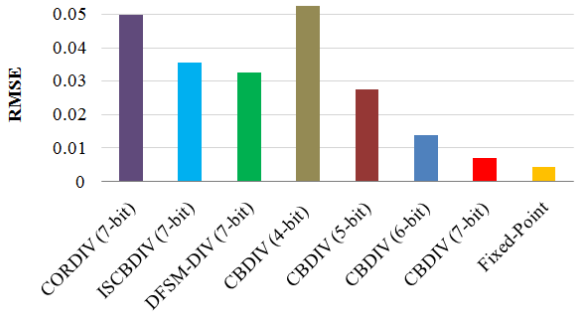

Fig. 8: Error behavior of CBDIV with different precision.

D. Comparison in an image process application

We implement the proposed CBDIV method for an image process application, $Contrast\ Stretch$ to further compare the performance against the baseline design and the state of art works.

The bit-stream length of CBDIV, ISCBDIV [12] and DFSM-DIV [14] implemented are all 256-bit, as the pixels in the input figure with JPEG format are 8-bit precision. The contrast-stretched pixel $G(x, y)$ of an arbitrary input image pixel $I(x, y)$ can be calculated as (2).

$$G(x,y) = \frac{I(x,y) - I_{min}}{I_{max} - I_{min}} \cdot 255 \tag{2}$$

Where I_{max} and I_{min} are the maximum and minimum pixel value in the image $I(x, y)$, respectively. Note that we first perform the division of $\frac{I(x,y)-I_{min}}{I_{max}-I_{min}}$, since SC dividers always require the dividend smaller than the divisor. We also utilize the CBSC method to do the multiplication illustrated in Sec. II-A. 255 can be treated as $255/256 \cdot 2^8$, then w in CBSC multiplier will be $255/256$ and the process requires 255 clock cycles to finish. According to the deterministic pattern used in CBSC method, all bit '1's in the divisor ($\frac{I(x,y)-I_{min}}{I_{max}-I_{min}}$) bit-stream will appear in these 255 clock cycles, meaning that we only need to count how many bit '1's are there in the divisor bit-stream. For CBDIV, the counting process is finished at the same time when CBDIV terminates. We don't require any more action and simply output the CBDIV result as the value of $G(x, y)$. While the outputs for ISCBDIV and DFSM-DIV are SN bit-streams, so we just add a counter to obtain $G(x, y)$.

We apply contrast stretching to several input images, using PSNR (Peak Signal Noise Ratio) to evaluate the output image quality. Contrast stretched images with an exact divider are used as the baseline. The PSNR values of all the tested images with different SC dividers are shown in Table II. We can observe that the proposed CBDIV method outperforms the two state of art SC dividers in all the tested images, and improves the image quality by 20.6 dB in average.

TABLE II: PSNR for contrast stretch application using SC dividers

	PSNR (dB)		
	ISCBDIV	DFSM-DIV	CBDIV
Map	34.84	36.28	**53.69**
Portrait	36.19	27.30	**57.40**
Woman	30.22	26.37	**53.11**
Lena	32.80	25.35	**54.62**

V. Conclusion

In this paper, we have proposed a fast and energy efficient approximate divider design based on stochastic computing division design, CBDIV, which exploits both the correlation requirement of existing SC-based division methods and the high efficiency of counting-based SC scheme. CBDIV fits well with the hybrid computing in which binary and SC implementations are both required for overall better application performance, like image processing. Experimental results showed that the proposed CBDIV outperformed state of art works by 77.8% in accuracy, 37.1% in delay, 21.5% in area, 50.6% in ADP and 25.9% in power. CBDIV also saves 31.9% in energy consumption when compared to the fixed-point division baseline, and is much more energy efficient than early proposed SC-based dividers if the inputs and outputs are both binary numbers. Furthermore, CBDIV with 5-bit precision can even outperform state of art works with 7-bit precision in accuracy by 15.4%. Finally, we compared CBDIV with other state of art SC dividers in contrast stretch application and showed that CBDIV could improve the accuracy with 20.6dB in average, which was a huge improvement.

References

[1] A. Naderi, S. Mannor, M. Sawan, and W. J. Gross, "Delayed stochastic decoding of ldpc codes," *IEEE Transactions on Signal Processing*, vol. 59, no. 11, pp. 5617–5626, 2011.

[2] P. Li, D. J. Lilja, W. Qian, K. Bazargan, and M. D. Riedel, "Computation on stochastic bit streams digital image processing case studies," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 3, pp. 449–462, 2013.

[3] K. Kim, J. Kim, J. Yu, J. Seo, J. Lee, and K. Choi, "Dynamic energy-accuracy trade-off using stochastic computing in deep neural networks," in *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, IEEE, 2016.

[4] H. Sim, D. Nguyen, J. Lee, and K. Choi, "Scalable stochastic-computing accelerator for convolutional neural networks," in *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 696–701, IEEE, 2017.

[5] H. Sim and J. Lee, "A new stochastic computing multiplier with application to deep convolutional neural networks," in *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, IEEE, 2017.

[6] R. Hojabr, K. Givaki, S. Tayaranian, P. Esfahanian, A. Khonsari, D. Rahmati, and M. H. Najafi, "Skippynn: An embedded stochastic-computing accelerator for convolutional neural networks," in *Proceedings of the 56th Annual Design Automation Conference 2019*, p. 132, ACM, 2019.

[7] S. Liu and J. Han, "Energy efficient stochastic computing with sobol sequences," in *Proceedings of the Conference on Design, Automation & Test in Europe*, pp. 650–653, European Design and Automation Association, 2017.

[8] S. Liu and J. Han, "Toward energy-efficient stochastic circuits using parallel sobol sequences," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 7, pp. 1326–1339, 2018.

[9] F. Neugebauer, I. Polian, and J. P. Hayes, "Building a better random number generator for stochastic computing," in *2017 Euromicro Conference on Digital System Design (DSD)*, pp. 1–8, IEEE, 2017.

[10] K. Kim, J. Lee, and K. Choi, "An energy-efficient random number generator for stochastic circuits," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 256–261, IEEE, 2016.

[11] T.-H. Chen and J. P. Hayes, "Design of division circuits for stochastic computing," in *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 116–121, IEEE, 2016.

[12] D. Wu and J. S. Miguel, "In-stream stochastic division and square root via correlation," in *Proceedings of the 56th Annual Design Automation Conference 2019*, p. 162, ACM, 2019.

[13] B. R. Gaines, "Stochastic computing systems," in *Advances in information systems science*, pp. 37–172, Springer, 1969.

[14] N. Temenos and P. P. Sotiriadis, "Deterministic finite state machines for stochastic division in unipolar format," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, IEEE, 2020.

[15] S.-I. Chu, Y.-M. Lee, C.-E. Hsieh, J.-H. Yen, and Y.-J. Huang, "Stochastic circuit design of image contrast stretching," in *2019 International SoC Design Conference (ISOCC)*, pp. 81–82, IEEE, 2019.

[16] R. Goldman, K. Bartleson, T. Wood, K. Kranen, V. Melikyan, and E. Babayan, "32/28nm educational design kit: Capabilities, deployment and future," in *2013 IEEE Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (PrimeAsia)*, pp. 284–288, IEEE, 2013.