This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2017.2789243, IEEE Transactions on Emerging Topics in Computing

High Quality Down-Sampling for Deterministic Approaches to Stochastic Computing

M. Hassan Najafi, *Student Member, IEEE,* David J. Lilja, *Fellow, IEEE*

Abstract—Deterministic approaches to stochastic computing (SC) have been recently proposed to remove the random fluctuation and correlation problems of SC and so produce completely accurate results with stochastic logic. For many applications of SC, such as image processing and neural networks, completely accurate computation is not required for all input data. Decision-making on some input data can be done in a much shorter time using only a good approximation of the input values. While the deterministic approaches to SC are appealing by generating completely accurate results, the cost of precise results makes them energy inefficient for the cases when slight inaccuracy is acceptable. In this work, we propose a high quality down-sampling method for previously proposed deterministic approaches to SC by generating pseudo-random – but accurate – stochastic bit streams. The result is a much better accuracy for a given number of input bits. Experimental results show that the processing time and the energy consumption of these deterministic methods are improved up to 61% and 41%, respectively, while allowing a mean absolute error (MAE) of 0.1%, and up to 500X and 334X improvement, respectively, for an MAE of 3.0%. The accuracy and the energy consumption are also improved compared to conventional random stream-based stochastic implementations.

Index Terms—Stochastic computing, high quality down-sampling, deterministic computing, energy-efficient processing, unary bit-stream, pseudo-randomized bit-stream.

1 Introduction

Stochastic Computing (SC) [1], [6], [21] has been around for many years as a noise-tolerant approximate computing approach. Logical computation is performed on probability data represented by uniformly distributed random bit-streams. Image and video processing [2], [5], [14], [20], digital filters [15], low-density parity check decoding [16], [22] and neural networks [3], [4], [9], [11], [12], [13] have been the main target applications for SC. Low hardware cost and low power consumption advantages of this computing paradigm have encouraged designers to implement complex calculations in the stochastic domain. Multiplication, for instance, can be implemented using a simple standard AND gate. Inherent skew tolerance [18] and progressive precision [2] are other interesting advantages of computation on stochastic bit-streams. Specifically, the computation results are correct even when the inputs are misaligned temporally, and the quality of the results improves as the computation proceeds. These two properties can be exploited in improving the processing speed of stochastic systems by optimizing clock distribution networks and making quick decisions on the input data.

Random Fluctuation, however, has always made stochastic computation somewhat inaccurate. Due to random fluctuation, stochastic operations often need to run for a very long time to produce highly accurate results. Recent progress in the idea of SC [17] [8], however, has revolutionized the paradigm and has changed the common belief on stochastic processing, that is, SC does not necessarily have to be an approximate computing approach. If properly structured, random fluctuation can be removed and SC circuits can produce deterministic and completely accurate results. Najafi et al [17] have shown that by choosing relatively prime lengths for a specific class of stochastic streams–called unary streams, and repeating the streams up to the least common multiple of the stream lengths, a deterministic and completely accurate output can be produced by stochastic logic. Jenson and Riedel [8] further proposed two deterministic approaches of processing unary streams, rotation of streams and clock division. The proposed approaches not only are able to produce completely accurate results (i.e., zero percent error rate), but they also improve the hardware cost and the processing time of stochastic operations significantly when compared to those of the computations performed on the conventional randomized stochastic bit-streams.

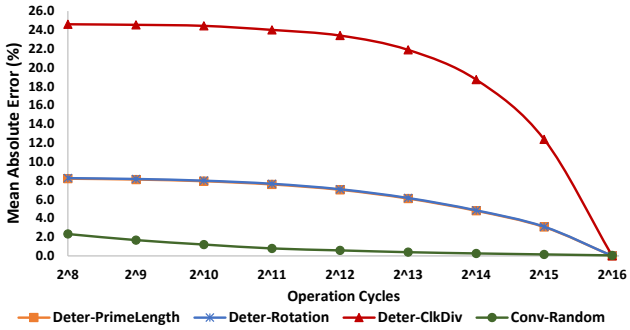

Fig. 1. Progressive Precision comparison of the conventional random stream-based SC with the unary stream-based deterministic approaches of SC when multiplying two 8-bit precision input values.

• *The authors are with the Department of Electrical and Computer Engineering, University of Minnesota, Twin Cities, MN 55455 USA (e-mail: najaf011@umn.edu; lilja@umn.edu).*

Manuscript received June 28, 2017; revised Dec 7, 2017.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2017.2789243, IEEE Transactions on Emerging Topics in Computing

While the unary stream-based deterministic approaches proposed in [17] and [8] are able to produce completely accurate results (i.e., results that are the same as the results of binary-radix computation), they suffer from a poor *progressive precision* property; The output converges to the expected correct value very slowly. This drawback can be a major limitation to wide use of these approaches in different applications. While ideally we are interested in producing completely accurate results, decision making on some inputs, particularly in image processing and neural network applications, do not require high precision operation and a low-precision estimate of the output value is sufficient. In such cases, due to the poor progressive precision property of unary streams, stochastic operations must run for a much longer time than the cases with conventional random bit-streams to produce acceptable results with small levels of inaccuracy. When small rates of inaccuracy are acceptable, using the unary stream-based deterministic approaches will lead to a very long operation time and consequently a very high energy consumption.

Fig. 1 compares the progressive precision of different stochastic approaches when multiplying 8-bit precision input values. As can be seen in the figure, the conventional random stream-based stochastic approach shows a much better progressive precision than the unary stream-based deterministic approaches and so is the preferable choice for any application that can tolerate some errors, such as image processing and neural network applications.

In this paper, we propose a down-sampling method for the deterministic approaches introduced in [17] and [8] that improves their progressive precision property. By modifying the structure of the stream generators, the deterministic methods not only are able to produce completely accurate results, they are also able to produce acceptable results in a much shorter time and so with a much lower energy consumption compared to the current architectures that generate and process unary streams. Our experimental results further show that, for the same operation time, deterministic down-sampling of the rotation and the relatively prime length approaches produces results with a lower average error rate than the error rate of processing conventional random stochastic streams.

This paper is structured as follows: Section 2 presents background information on SC including the conventional random stream-based approach and the deterministic approaches to SC. In Section 3, we describe our proposed down-sampling method. In Section 4, we validate and compare the proposed idea to the prior methods with experimental results. Finally, in Section 5, we present conclusions.

2 Background

2.1 Stochastic Computing

In SC, computation is performed on random [21] or unary bit-streams [19] [17] [8] where the input value is encoded by the probability of obtaining a one versus a zero. Unipolar and bipolar formats are the two general representations for numbers in the stochastic domain. While the unipolar format can only be used for representing positive data in interval [0,1], the bipolar format can deal with both positive and negative values in [-1, 1]. In the unipolar representation,

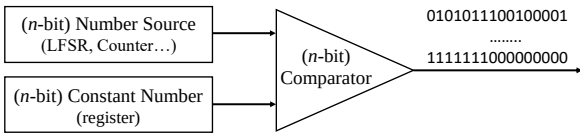

Fig. 2. Structure of a stochastic stream generator.

the ratio of the number of ones to the length of bit stream determines the value, while in the bipolar format, the value is determined by the difference between the number of ones and zeros compared to the stream length. For example, 1101010000 is a representation of 0.4 in the unipolar format and -0.2 in the bipolar format. For the rest of the paper we use the unipolar format. However, the proposed idea is independent of the format of bit-streams and can also be applied to the bipolar representation.

The inputs to stochastic systems must first be converted to stochastic bit-streams to be processed by stochastic logic. The common approach for converting digital data in binary radix format into random stochastic bit-streams is by comparing a random value generated by a random or pseduo-random source to the target value. Linear feedback shift registers (LFSRs) are often used as the pseudo-random source in these stream generators. To convert binary input data to unary streams, an increasing/decreasing value from an up/down counter is compared to the target value. Fig. 2 shows the structure of these basic stochastic stream generators.

When performing computation on random stochastic bit-streams, due to the inherent random fluctuations, the length of bit-streams have to be much longer than the precision expected for the computation result. Some operations, such as multiplication, also suffer from correlation between bit-streams. For these operations, the input bit-streams must be independent to produce accurate results. To produce an output with n-bit precision, the input bit-streams length, and so the number of cycles performing the operation, must be greater than 2^{2ni-2}, where i is the number of independent inputs in the circuit [8]. Due to these properties, stochastic processing of random bit-streams is an approximate computation, as illustrated in Fig. 3.a.

2.2 Deterministic Approaches to Stochastic Computing

Recent work on SC [17] [8] [19] has shown that SC does not necessarily have to be an approximate approach and the result of computation can actually be completely accurate and deterministic. Instead of random stochastic bit-streams, logical computation is performed on a specific class of bit-streams, called unary streams. A unary stream consists of a sequence of 1s followed by a sequence 0s. For example, 1111000000 is a unary stream representing 0.4 in the unipolar format. To represent a value with resolution of $1/2^n$ (n-bit precision), the unary stream must be 2^n bits long. For operations that require independent inputs, the independence between the input unary streams is provided by using relatively prime stream lengths [17], rotation, or clock division [8]. Fig. 3(b)-(d) exemplifies these three deterministic approaches to SC.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2017.2789243, IEEE Transactions on Emerging Topics in Computing

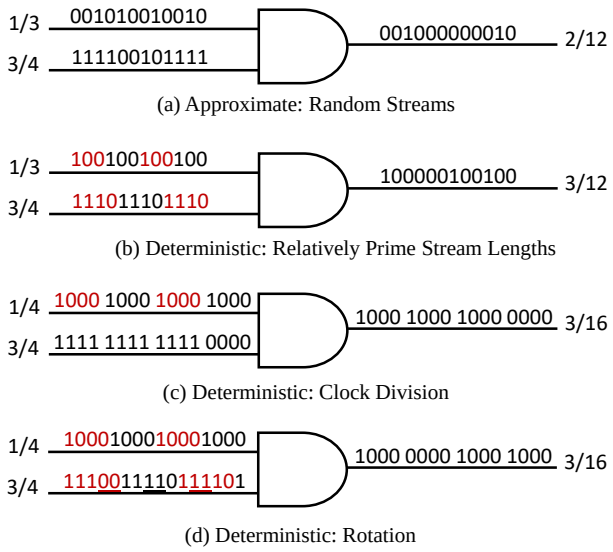

Fig. 3. Examples of performing stochastic multiplication: a) conventional approximate SC with random bit-streams (b)-(d) recently proposed deterministic approaches to SC with unary bit streams.

To produce accurate results with these deterministic approaches, the operation must run for an exact number of clock cycles which is equal to the product of the length of the input bit streams. For example, when multiplying two n-bit precision input values represented using two 2^n-bit streams, the operation must run for exactly 2^{2n} cycles [8]. Running the operation for fewer cycles (e.g., 2^{2n-1} cycles) will lead to a poor result with an error out of the acceptable error bound. This important source of inaccuracy in performing computations on unary streams is called "truncation error" [17].

As an example, assume we want to multiply two 8-bit precision numbers, represented using unary streams, with the rotation or clock division deterministic approaches. The operation must run for exactly $2^{16} = 65536$ cycles to produce a completely accurate result. Exhaustively testing the multiplication operation for every possible pair of input values when running the operation for 2^{15} and 2^{10} cycles shows a mean absolute error (MAE) of 3.10% and 7.99%, respectively, for the rotation approach, and 12.3% and 24.4% for the clock division approach. With the conventional approach of processing random bit-streams when exhaustively testing the operation on a large set of random pairs of input values, although we could not produce completely accurate multiplication results in 2^{16} cycles, a good progressive precision property could lead to acceptable results when running the operation for the same number of operation cycles (MAE of 0.15% after 2^{15} and 1.20% after 2^{10} cycles).

3 Down-Sampling for Deterministic SC

While the randomness inherent in stochastic bit-streams was one of the main sources of inaccuracy in SC, distributing the ones across the stream instead of grouping them (i.e., first all ones and then all zeros) may be able to provide a good progressive precision property for representing stochastic numbers and, therefore, for the computation. With randomized bit-streams the result quality improves as the computation

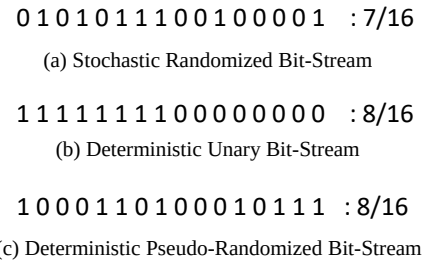

Fig. 4. Different types of stochastic bit-streams.

proceeds. This is because short sub-sequences of long random stochastic bit-streams provide low-precision estimates of the streams' values. This property can be exploited in many applications of SC for making quick decisions on the input data and so increasing the processing speed.

Deterministic approaches proposed in [17] and [8] perform computation on unary streams. Due to the nature of unary representation, truncating the bit-stream leads to a high truncation error and so a significant change in the represented value. In this work, we propose a high quality down-sampling approach for the deterministic approaches to SC by bringing randomization back into the representation of bit-streams. Similar to processing unary streams, the computations are completely accurate when the operations are executed for the required number of cycles. However, by pseudo-randomizing the streams, the computation will have a good progressive precision property and truncating the output streams by running for fewer clock cycles still produces high quality outputs.

For a deterministic and predictable randomization of the bit-streams, we propose to use *maximal period* pseudo-random sources (i.e., a maximal period LFSR) to generate the bit-streams. The important point is that the period of the pseudo-random source should be equal to the length of the bit-stream. By using such a source to generate random numbers, we are able to convert an input value into a pseudo-random but completely accurate stochastic representation. Fig.4 illustrates an example of representing 0.5 value with a random, a unary, and our proposed pseudo-randomized bit-stream.

Table 1 compares the MAEs of the conventional random stream-based SC and the unary stream-based deterministic approaches of [17] and [8] with the approach proposed in this work by exhaustively testing multiplication of two 8-bit precision stochastic streams on a large set of random input values for the conventional random SC and for the proposed approach, and on every possible input value for the unary deterministic approaches. For the conventional random stochastic approach, we evaluate the accuracy with two different structures for converting the input values to randomized stochastic bit-streams: 1) using maximal period 8-bit LFSRs, and 2) using maximal period 16-bit LFSRs to emulate a true-random number generator. Two different LFSRs (i.e., different designs[1] with different seeds) are used in each case to generate independent bit-streams. While the first structure can accurately convert the input values to

1. Two out of 16 different designs of maximal period 8-bit LFSRs and two out of 2,048 different designs of maximal period 16-bit LFSRs described in [10] are randomly selected for each run.

TABLE 1
Mean Absolute Error (%) comparison of the prior random and deterministic approaches to stochastic computing and the proposed deterministic approaches based on pseudo-randomized streams when multiplying two 8-bit precision stochastic streams with different numbers of operation cycles.

Design Approach	SNG	2^{16}	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}
Conventional Random Stochastic	Prior work [1], [21]: two LFSR-8	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	2.54	4.25
	Prior work [1], [21]: two LFSR-16	0.05	0.15	0.26	0.39	0.58	0.79	1.20	1.67	2.32	3.32	4.72
Deterministic Prime Length	Prior work [8], [17]: two counter-8	0.00	3.03	4.70	6.01	7.08	7.62	7.90	7.98	8.11	33.2	51.5
	This work: two LFSR-8	0.00	0.09	0.16	0.24	0.34	0.47	0.60	0.72	0.85	2.56	4.22
Deterministic Clock Division	Prior work [8]: two counter-8	0.00	12.3	18.7	21.8	23.4	24.0	24.4	24.5	24.9	49.6	62.2
	This work: two LFSR-8	0.00	1.44	2.48	3.74	5.28	7.18	9.91	14.2	24.8	25.0	25.8
Deterministic Rotation	Prior work [8]: two counter-8	0.00	3.10	4.84	6.15	7.08	7.66	7.99	8.17	8.26	33.1	51.8
	This work: two LFSR-8	0.00	0.09	0.16	0.24	0.35	0.47	0.60	0.71	0.82	2.56	4.26

256-bit[2] pseudo-random bit-streams, the second structure converts the inputs to any stream with a length less than 2^{16} to give an approximate representation of the value. With the first structure, after 256 cycles, the generated bit-streams repeat and so the accuracy of the operation never improves after this time. Due to a more precise representation, the first structure shows a better MAE for low stream lengths. However, for very long bit-stream lengths, the second structure can produce a better MAE. The hardware cost of the second structure is twice that of the first one because of using larger LFSRs. Note that due to random fluctuation and correlation, neither of these two structures can produce completely accurate results in 2^{16} cycles.

As shown in Table 1, the deterministic approaches proposed in [17] and [8] are able to produce completely accurate results when running the operation for 2^{16} cycles. Due to using unary bit-streams, however, the MAE of the computation increases significantly when running the operation for fewer cycles. This change clearly shows the poor progressive precision property and the high truncation error of these methods. Instead of unary streams, in this work we use pseudo-randomized but accurate bit-streams. Integrating these bit-streams with the deterministic approaches results in completely accurate computation when it is run for the required number of cycles while still producing high quality results if the output stream is truncated.

As discussed in Section 2, in the deterministic approaches to SC, the required independence between input streams is provided by using relatively prime lengths, rotation, or clock division. When running the operations for the product of the length of the streams, these three methods cause *every bit of the first stream to interact with every bit of the second stream* [8], [17]. The computation is therefore performed deterministicly and accurately irrespective of the location of the ones in each stream. Thus, as demonstrated in Fig. 5 with the interaction of two pseudo-randomized bit-streams, there is actually no requirement to use unary-style streams and, instead, we use pseudo-randomized bit-streams for the deterministic approaches.

We use different LFSRs (different LFSR designs and different seeds) for generating pseudo-randomized bit-streams. The period of the LFSR should be maximal and

2. An n-bit maximal period LFSR has a period of $2^n - 1$, as the 0-state in the LFSR is normally not used. Here, for a fair comparison with the unary stream-based deterministic approaches, we add a 0-state to the set of the states of each LFSR to generate 2^n unique numbers.

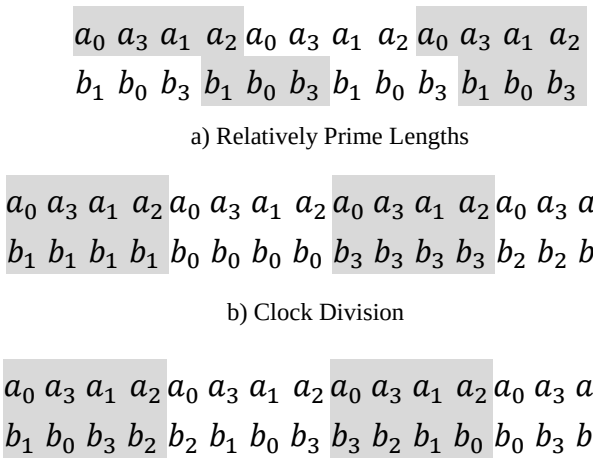

Fig. 5. Deterministic approaches to SC by two pseudo-randomized bit-streams.

equal to the length of the bit-stream to accurately represent each value. Thus, for 8-bit precision inputs, an 8-bit size maximal period LFSR is required. Table 1 compares the MAE of the deterministic approaches when multiplying the inputs streams generated using the proposed approach. Similar to the unary stream-based deterministic approaches of [8], the proposed method results in completely accurate results when running the operation for 2^{16} cycles, but produce a much lower MAE when running for fewer cycles. Compared to the conventional random SC, the relatively prime length and the rotation approaches produce results with a lower MAE.

Note that, similar to the unary-stream based deterministic approaches that require n separate counters for generating n independent input bit-streams [8], sharing LFSRs in the proposed method is not possible. In the clock division deterministic approach, each LFSR must be driven with a different clock source which as a result prevents using optimization techniques such as sharing LFSRs+shifting [7] to save hardware cost. Similarly, the limitation of using number sources with different periods in the relatively prime approach and stalling number generators in the rotation approach prevent us from sharing pseudo-random number generators in the proposed method.

4 Experimental Results

To evaluate the proposed idea, we used the stochastic implementation of a well-known digital image processing algorithm, Robert's cross edge detection. In this edge detector, each operator consists of a pair of 2×2 convolution kernels that process the pixels of the input images based on their three immediate neighbors:

$$Y_{i,j} = 0.5 \times (|X_{i,j} - X_{i+1,j+1}| + |X_{i,j+1} - X_{i+1,j}|)$$

where $X_{i,j}$ is the value of the pixel at location (i, j) of the input image and $Y_{i,j}$ is the corresponding output value. Fig. 6 shows the stochastic implementation of this algorithm proposed in [2].

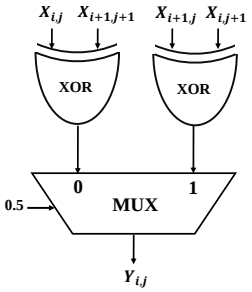

Fig. 6. Stochastic circuit for Robert's cross edge detection algorithm [2].

The two XOR gates compute absolute value subtraction when they are fed with correlated input streams (streams with maximum overlap between 1s). Sharing the same source of numbers (i.e., same LFSR) for generating the input streams can provide correlated streams. The Multiplexer (MUX) unit, on the other hand, performs scaled addition irrespective of correlation between its main input streams. The important point, however, is that the select input stream (here, a stream with the value 0.5) should be independent to the main input streams to the MUX. Thus, for the Robert's cross stochastic circuit, the four main input streams (the inputs to the XOR gates) should be correlated to each other, but should be independent of the select input of the MUX. Two number generators are, therefore, required for this circuit – one for converting the main inputs and one for generating the select input stream.

We evaluate the performance, the hardware area, the power, and the energy consumption of the Robert's cross stochastic circuit in three different cases: 1) the conventional approach of processing random streams, 2) the prior deterministic approaches of processing unary streams, and 3) the proposed deterministic approaches of processing pseduo-randomized streams. The circuit shown in Fig. 6 is the core stochastic logic and will be shared between all cases.

Fig. 7 shows our proposed structures of the sources for generating pseudo-random numbers for the three deterministic approaches. For the relatively prime length approach, we assume the first number source has a period of $2^n - 1$and we control the period of the second source by setting a stop state. Here, for the Robert's cross circuit, pseudo-random number sources with periods of $2^8 - 1$ and $2^8 - 2$, are implemented. When the state (the output number) of LFSR 2 equals the stop state, LFSR 2 is restarted to its initial state.

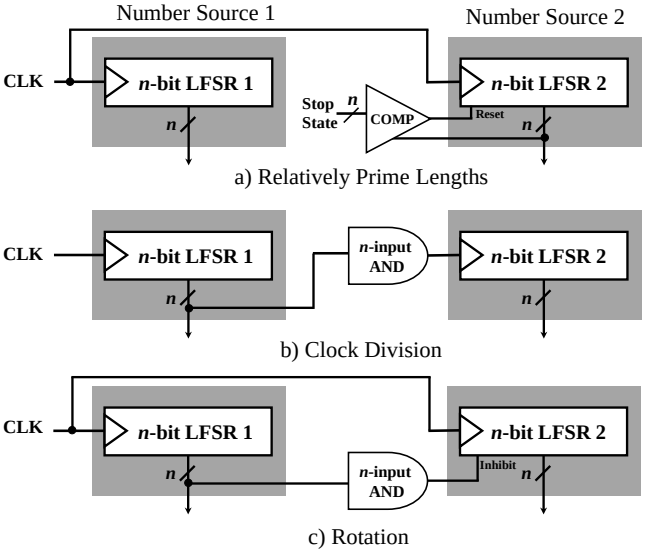

Fig. 7. Proposed sources of generating pseudo-random numbers for the three deterministic approaches to SC.

For the clock division structure, LFSR 2 is clock divided by the period of LFSR 1 through detecting the all one state using an AND gate. Similarly, the rotation structure uses an AND gate to inhibit or stall every $2^n - 1$ cycles when the all one state is detected.

These units are used as the number source in the stochastic stream generator shown in Fig. 2. For the unary stream-based prior deterministic approaches, we optimized and implemented the counter-based architectures of [8]. For the conventional random stream-based implementations, we used two different 8-bit or two different 16-bit LFSRs as the required sources of random numbers. We used the Synopsys Design Compiler vH2013.12 with a 45nm gate library to synthesize the designs.

As shown in Table 2, the hardware area cost of the proposed deterministic designs is slightly (<10%) more than that of their corresponding prior deterministic implementations. Due to replacing counters with LFSRs in the proposed architectures, the power consumption has also increased in all cases. The important metric, however, in evaluating the efficiency of the implemented designs is energy consumption, defined as the product of the power consumption and processing time.

We evaluate the energy-efficiency of the different designs by measuring the energy consumption of each one in achieving a specific accuracy in processing the inputs. MAE is used as the accuracy metric (a lower MAE means a higher accuracy). To comprehensively test the designs, we simulate the operation of the Robert's cross circuit in each design approach by processing 10,000 sets of 8-bit precision random input values. For accurate representation of input values in each design approach, we randomly choose an integer value between zero and the period of the (pseudo-random) number generator and divide it by the period. Fig. 8 and Fig. 9 present the MAE and the standard deviation of processing random input values in different design approaches. Table 2 further shows the number of processing cycles and the energy consumption of each design to achieve different accuracies.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2017.2789243, IEEE Transactions on Emerging Topics in Computing

TABLE 2
Area (um^2), Power (mW) (@ 1GHz), and Energy consumption (pJ) of the Robert's cross stochastic circuit synthesized with the 8- and 16-bit conventional random approach, and also the prior structures and the proposed structures of the three deterministic approaches (Relatively Prime Lengths, Clock Division, and Rotation).

			Target Error (MAE)													
Design	Area	Power	0%		0.1%		0.3%		0.5%		1.0%		2.0%		3.0%	
Approach	(μm^2)	(mW)	Cycle	Energy	Cycle	Energy	Cycle	Energy	Cycle	Energy	Cycle	Energy	Cycle	Energy	Cycle	Energy
Conv-Random-8	671	0.818	-	-	-	-	-	-	-	-	-	-	170	**139**	100	**82**
Conv-Random-16	1415	1.633	2^16>	10^5>	54410	88,852	11370	18,567	4380	7,153	1220	1,992	300	490	130	212
PrimeL-Prior	689	0.560	64770	**36,271**	64210	35,958	63150	35,364	62070	34,759	59510	33,326	54420	30,475	49080	27,485
PrimeL-Proposed	746	0.848	64770	54,925	25100	**21,285**	4000	**3,392**	1500	**1,272**	470	**399**	170	**144**	100	**85**
CLKDIV-Prior	665	0.304	65025	**19,768**	64830	**19,708**	63850	19,410	62900	19,122	60740	18,465	56740	17,249	53250	16,188
CLKDIV-Proposed	688	0.527	65025	34,268	63730	33,586	54310	28,621	42640	22,471	20780	10,951	6500	3,426	2980	1,570
Rotation-Prior	667	0.493	65025	**32,057**	64665	31,880	63445	31,278	62405	30,766	59855	29,509	54875	27,053	49575	24,440
Rotation-Proposed	725	0.732	65025	47,598	29305	**21,451**	4045	**2,961**	1495	**1,094**	465	**340**	170	**124**	100	**73**

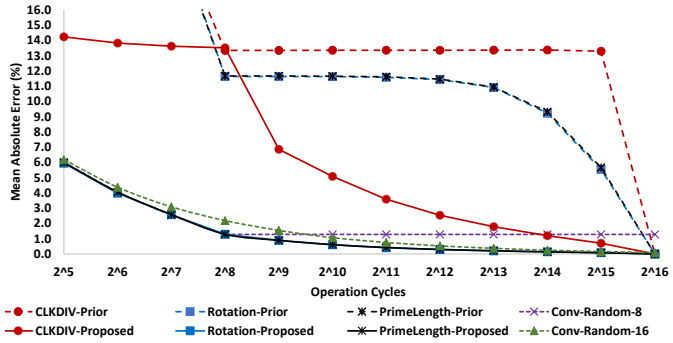

Fig. 8. Mean Absolute Error (%) when processing random input values with the Robert's cross stochastic circuit using different stochastic approaches.

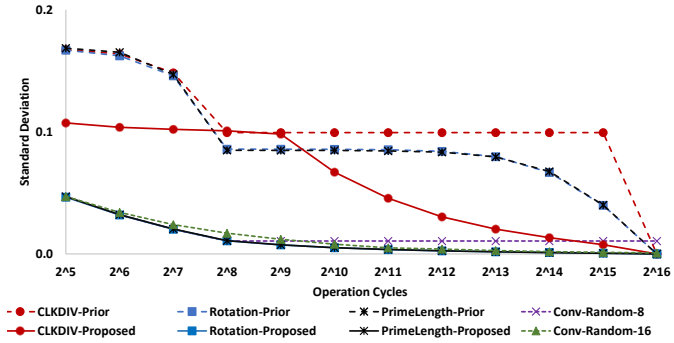

Fig. 9. Standard deviation of the absolute error of processing random input values with the Robert's cross stochastic circuit using different stochastic approaches.

When completely accurate results are expected, the proposed designs must run for the same number of cycles as required by the prior deterministic designs (product of the periods of the number generators). Considering the higher power consumption of the proposed designs, the prior deterministic implementations consume less energy to achieve completely accurate results. The great advantage of the proposed architectures starts when slight inaccuracy in the computation is acceptable. In such cases, the proposed designs start showing a much lower energy consumption by converging to the expected accuracy in a much shorter time.

For the relatively prime and the rotation approaches, the proposed designs improve the processing time by 61% and 55%, respectively, resulting in an energy consumption savings of 41% and 33% when accepting an MAE of as low as 0.1%. For an MAE of 3.0%, these architectures consume 324 and 334 times lower energy by improving the processing time by up to 500X compared to prior architectures of [8]. For the clock division approach, the proposed design is more energy efficient if at least an MAE of 1.0% is acceptable. The energy consumption is reduced 10 times for this method for an MAE of 3%.

Compared to the conventional random stream-based architectures (Conv-Random-8 with 8-bit LFSRs and Conv-Random-16 with 16-bit LFSRs) the proposed structures are more energy-efficient than the 16-bit conventional architecture but are at the same level with the 8-bit implementation. The important point, however, is that the 8-bit conventional architecture cannot achieve an MAE of 1.0% or lower and the 16-bit architecture requires a very long processing time and consumes significant energy to get close to completely accurate results.

5 Conclusion

Recent work on SC has shown that computation using stochastic logic can be performed deterministically and accurately by properly structuring unary-style bit-streams. The hardware cost and the latency of operations are much lower than those of the conventional random SC when completely accurate results are expected. For applications that slight inaccuracy is acceptable, however, these unary stream-based deterministic approaches must run for a relatively long time to produce acceptable results. This processing time, which is often much longer than the latency of the conventional random SC in achieving the same accuracy levels, makes the deterministic approaches energy-inefficient.

While randomness was a source of inaccuracy in the conventional random stream-based SC, we exploited pseudo-randomness in improving the progressive precision prop-

erty of the deterministic approaches to SC. Completely accurate results are still produced if running the operation for the required number of cycles. When slight inaccuracy is acceptable, however, significant improvement in the processing time and energy consumption is observed compared to the prior unary stream-based deterministic approaches and also the conventional random-stream based approaches. The proposed approach is applicable to any operation covered by the deterministic approaches of SC.

ACKNOWLEDGMENTS

This work was supported in part by National Science Foundation grant no. CCF-1408123. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

REFERENCES

[1] A. Alaghi and J. P. Hayes. Survey of stochastic computing. *ACM Trans. Embed. Comput. Syst.*, 12(2s):92:1–92:19, 2013.

[2] A. Alaghi, C. Li, and J. Hayes. Stochastic circuits for real-time image-processing applications. In *Design Automation Conference (DAC), 2013 50th ACM / EDAC / IEEE*, pages 1–6, May 2013.

[3] A. Ardakani, F. Leduc-Primeau, N. Onizawa, T. Hanyu, and W. J. Gross. Vlsi implementation of deep neural network using integral stochastic computing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, PP(99):1–12, 2017.

[4] B. Brown and H. Card. Stochastic neural computation. i. computational elements. *Computers, IEEE Transactions on*, 50(9):891–905, Sep 2001.

[5] D. Fick, G. Kim, A. Wang, D. Blaauw, and D. Sylvester. Mixed-signal stochastic computation demonstrated in an image sensor with integrated 2d edge detection and noise filtering. In *Proceedings of the IEEE 2014 Custom Integrated Circuits Conference*, pages 1–4, Sept 2014.

[6] B. Gaines. Stochastic computing systems. In *Advances in Information Systems Science*, Advances in Information Systems Science, pages 37–172. Springer US, 1969.

[7] 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)*, pages 361–366, Oct 2014.

[8] D. Jenson and M. Riedel. A deterministic approach to stochastic computation. In *Proceedings of the 35th International Conference on Computer-Aided Design*, ICCAD '16, pages 102:1–102:8, New York, NY, USA, 2016.

[9] 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 *Proceedings of the 53rd Annual Design Automation Conference*, DAC '16, pages 124:1–124:6, New York, NY, USA, 2016. ACM.

[10] P. Koopman. Maximal Length LFSR Feedback Terms https://users.ece.cmu.edu/ koopman/lfsr/index.html, 2017.

[11] V. T. Lee, A. Alaghi, J. P. Hayes, V. Sathe, and L. Ceze. Energy-efficient hybrid stochastic-binary neural networks for near-sensor computing. In *Proceedings of the Conference on Design, Automation & Test in Europe*, DATE '17, pages 13–18, 3001 Leuven, Belgium, Belgium, 2017. European Design and Automation Association.

[12] B. Li, M. H. Najafi, and D. J. Lilja. Using stochastic computing to reduce the hardware requirements for a restricted boltzmann machine classifier. In *Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '16, pages 36–41, New York, NY, USA, 2016. ACM.

[13] B. Li, Y. Qin, B. Yuan, and D. J. Lilja. Neural network classifiers using stochastic computing with a hardware-oriented approximate activation function. In *2017 IEEE International Conference on Computer Design (ICCD)*, pages 97–104, Nov 2017.

[14] P. Li, D. Lilja, W. Qian, K. Bazargan, and M. Riedel. Computation on stochastic bit streams digital image processing case studies. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 22(3):449–462, 2014.

[15] Y. Liu and K. K. Parhi. Architectures for recursive digital filters using stochastic computing. *IEEE Transactions on Signal Processing*, 64(14):3705–3718, July 2016.

[16] A. Naderi, S. Mannor, M. Sawan, and W. Gross. Delayed stochastic decoding of ldpc codes. *Signal Processing, IEEE Transactions on*, 59(11):5617–5626, Nov 2011.

[17] M. H. Najafi, S. Jamali-Zavareh, D. J. Lilja, M. D. Riedel, K. Bazargan, and R. Harjani. Time-encoded values for highly efficient stochastic circuits. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 25(5):1–14, 2017.

[18] M. H. Najafi, D. J. Lilja, M. D. Riedel, and K. Bazargan. Polysynchronous clocking: Exploiting the skew tolerance of stochastic circuits. *IEEE Transactions on Computers*, 66(10):1734–1746, Oct 2017.

[19] M. H. Najafi, D. J. Lilja, M. D. Riedel, and K. Bazargan. Power and Area Efficient Sorting Networks using Unary Processing. In *Computer Design (ICCD), 2017 IEEE 35th International Conference on*, 2017.

[20] M. H. Najafi and M. E. Salehi. A Fast Fault-Tolerant Architecture for Sauvola Local Image Thresholding Algorithm using Stochastic Computing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(2):808–812, Feb 2016.

[21] W. Qian, X. Li, M. Riedel, K. Bazargan, and D. Lilja. An architecture for fault-tolerant computation with stochastic logic. *Computers, IEEE Trans. on*, 60(1):93–105, Jan 2011.

[22] S. Tehrani, S. Mannor, and W. Gross. Fully parallel stochastic ldpc decoders. *Signal Processing, IEEE Transactions on*, 56(11):5692–5703, Nov 2008.

M. Hassan Najafi (S'15) received the B.Sc. degree in computer engineering from University of Isfahan, Isfahan, Iran, and the M.Sc. degree in computer architecture from University of Tehran, Tehran, Iran, in 2011 and 2014, respectively. He is currently pursuing the Ph.D. degree with ARCTIC Labs, Department of Electrical and Computer Engineering, University of Minnesota, Twin cities, MN, USA. His current research interests include stochastic and approximate computing, computer-aided design of integrated circuits, low-power design, and designing fault tolerant systems. In recognition of his research, he received the Doctoral Dissertation Fellowship at the University of Minnesota and the Best Paper Award at the 2017 35th IEEE International Conference on Computer Design.

David J. Lilja (F06) received the B.S. degree in computer engineering from Iowa State University in Ames, IA, USA, and the M.S. and Ph.D. degrees in electrical engineering from the University of Illinois at Urbana-Champaign in Urbana, IL, USA. He is currently the Schnell Professor of Electrical and Computer Engineering at the University of Minnesota in Minneapolis, MN, USA, where he also serves as a member of the graduate faculties in Computer Science, Scientific Computation, and Data Science. Previously, he served ten years as the head of the ECE department at the University of Minnesota, and worked as a research assistant at the Center for Supercomputing Research and Development at the University of Illinois, and as a development engineer at Tandem Computers Incorporated in Cupertino, California. He was elected a Fellow of the Institute of Electrical and Electronics Engineers (IEEE) and a Fellow of the American Association for the Advancement of Science (AAAS).