Accurate Synthesis of Arithmetic Operations with Stochastic Logic

Ensar Vahapoglu, Mustafa Altun
Electronics and Communication Engineering Department
Istanbul Technical University
Istanbul, Turkey
{vahapoglu, altunmus}@itu.edu.tr

***Abstract*— In this study, we propose a method to overcome the main drawback in stochastic computing, low accuracy or related long computing times. Our method exploits dependency in stochastic bit streams with the aid of feedback mechanisms. Accurate (error-free) arithmetic multiplier and adder circuits are implemented. Operations are performed using both stochastic and binary inputs/outputs; binary-stochastic number conversion circuits are implemented for this purpose. We test our circuits by considering performance parameters area, delay, and accuracy. The simulation results are evaluated in a comparison with the results of other stochastic and deterministic (conventional) computing techniques in the literature. Additionally, we discuss the applicability of our method in emerging technologies including printed/flexible electronics for which low transistor counts is desired.**

Keywords*— *Stochastic computing, arithmetic circuits, accuracy, performance optimization, printed electronics

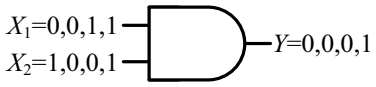
I. INTRODUCTION

While an *n*-bit binary number is represented using *n* separate routes/paths in conventional computing, stochastic computing can use a single path, independent of the bit count *n*, to represent the number in the form of a bit stream. This property was the main motivation for the birth of stochastic computing [1, 2], and provides an important circuit size reduction. For example, an *n*×*n*-bit stochastic multiplier can always be implemented with a single AND gate that is independent of *n*. In contrary, the circuit size of a conventional multiplier quickly grows with *n*; 4×4-bit and 8×8-bit binary deterministic multipliers require 100s and 1000s of gates, respectively. Despite this important advantage, stochastic computing could not become a real competitor to conventional computing except for few image processing applications not requiring high accuracy [3, 8]. Low accuracy and related long computing times are the main obstacles in front of stochastic computing and the main motivation of this study. In this paper, accurate and high performance stochastic arithmetic circuits including multipliers and adders are implemented.

Stochastic multiplication with a single AND gate is illustrated in Fig. 1. Here, the input and the output values are probabilities derived as the number of 1 valued bits over the total number of bits in a stream. In this figure the multiplication is accurate or error-free, but this is not guaranteed for different cases since 1s and 0s in a stream are randomly positioned in stochastic computing. Fig. 2 shows an example with an erroneous result for which expected output value 1/4 is not same as the real obtained value 0/4. Here, the relative standard error is 100% by using (1) below.

$$E = \frac{|(\text{real value}) - (\text{expected value})|}{(\text{expected value})} . \quad (1)$$

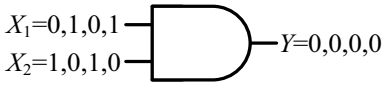
To improve accuracy or decrease error rates, the number of bits in streams can be increased, but this solution is not efficient at all. As shown in Fig. 3, to achieve error rates below 1%, streams having more than 1000 bits are needed that is not practical in terms of the computing time. Indeed, if input bit streams are randomly and independently generated that is the definition of conventional stochastic computing, then error rates at the outputs can be decreased only with increasing the number of bits in streams. At this point there emerge two probable solutions at the cost of worsening stochasticity. These are "decreasing randomness in input bit streams" and "using dependent/correlated bit streams".



$X_1 \times X_2 = Y$

$1/2 \times 1/2 = 1/4$

Fig. 1. Accurate stochastic multiplication with an AND gate



$X_1 \times X_2 = Y$

$1/2 \times 1/2 \neq 0$

Fig. 2. Inaccurate stochastic multiplication with an AND gate

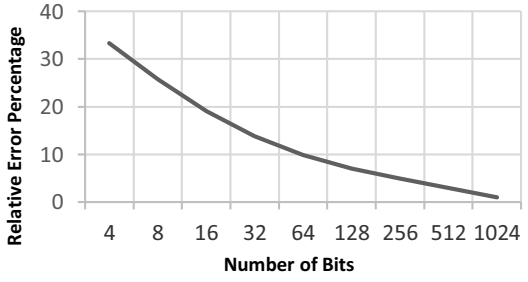


Fig. 3. Average error percentage for an AND gate with respect to the number of bits in streams; inputs X_1 and X_2 are 1/2.

978-1-4673-9039-2/16 $31.00 © 2016 IEEE
DOI 10.1109/ISVLSI.2016.74

Corresponding to the first solution, pseudo-random and quasi-random number generators are proposed in the literature [7, 8, 18]. Pseudo-random generators generally use LFSRs (Linear Feedback Shift Register) that even allow to produce desired orders of 0s and 1s resulting in perfect accuracy [4, 7]. Additionally, quasi-random generators producing low-discrepancy bit streams can decrease error rates [5]. The main disadvantage for these methods is considerably large circuit size. The size of the generators quickly grows with the number of bits and the number of inputs. This certainly kills the main advantage of stochastic computing "using small circuits and small number of transistors/gates". Our method satisfies perfect accuracy with smaller circuit sizes and computing times (delays). Our method corresponds to the second mentioned solution "using dependent/correlated bit streams". To the best of our knowledge there are two recent studies exploiting correlation for accuracy [9, 17]. These studies analyze correlated bit streams and formulize the necessary conditions to improve accuracy. However, they lack of offering circuit design techniques without a need of random number generators. Also they do not result in error-free outputs. Contrary to these methods, we achieve accuracy with keeping the circuit size still much smaller than those for conventional computing.

The proposed method is exclusively applicable for technologies necessitating low transistor counts. In this regard, printed/flexible electronics is a strong candidate. Conventional circuit design techniques are not proper for printed electronics since they necessitates relatively high number of transistors. Compared to CMOS transistors, printed organic thin film transistors (OTFT) have much larger dimensions and lower performances; circuits with thousands of transistors cannot be practically implemented with printed electronics [6, 19]. Printed circuits should have relatively small number of transistors; the proposed stochastic circuit blocks is quite suitable for this.

The organization of the paper is as follows. In Section II, we present our methodology. Subsections A and B explain our circuit implementation techniques for multiplication and scaled addition operations, respectively. In Section III, we present simulation results for the proposed multiplier circuit in comparison with the results in the literature by considering the performance parameters area, delay, and accuracy. Finally, in Section IV we overview our proposed methodology and discuss extensions and future directions for this research.

II. The Proposed Methodology

Improving accuracy in stochastic computing has some limits as explained in the theorem below.

Theorem: *If operations in a stochastic system are performed using randomly generated and uncorrelated bit streams that is the definition of conventional stochastic computing, then the accuracy of the system only depends on the expected output value z_e and the number of bits $\boldsymbol{n}$ in the output stream.*

Proof*: For the explained case, z_e can also be defined as the probability that each output bit takes logic 1 value. Therefore, each output bit has a Bernoulli distribution and the output stream has Binomial distribution ($p=z_e$). The standard error (standard deviation) and the relative error can be calculated as $\sqrt{\frac{p(1-p)}{n}}$ and $\sqrt{\frac{(1-p)}{pn}}$, respectively; both only depend on $p=z_e$ and $\boldsymbol{n}$.* □

This relatively simple theorem tells us:

- Circuit architecture or size does not affect the accuracy. For example, circuits with 1 gate and 1000 gates performing the same operation always have the same accuracy;
- We cannot improve accuracy unless we use correlated bit streams and/or generate non-random bit streams; and
- We cannot achieve perfect accuracy (error-free) unless we use correlated bit streams.

We exploit the above facts to implement error-free multiplier and adder circuits. We make manipulations in input bit streams to satisfy correlations. We also use feedback mechanisms for circuit size reduction that allows us to use the streams multiple times. For example, while copying n bits requires n memory elements in conventional computing, we achieve this with a single delay block thanks to the bit streams flowing in the time domain.

A. Multiplicaiton

To achieve accurate multiplication, we first present a necessary condition with the following lemma.

Lemma-1: *Consider m input bit streams, each having n bits with taking values between 1/n and n/n. Accurate stochastic multiplication of these inputs requires an output bit stream having at least n^m bits.*

Proof: *Consider the worst case scenario for which each and every input takes the value of 1/n. The output value should be $1/n^m$ that requires n^m bits.* □

Lemma-1 tells us that to perform error-free stochastic multiplication of two n-bit inputs, we optimally need an n^2-bit output. We satisfy this WİTH the circuit shown in Fig. 4. In order to prevent any information loss that results in inaccuracy, we need to account each and every bit in input bit streams, so each bit in one of the input streams should be ANDed with each bit in the other input stream. For this purpose we use the repeating and the bit shifting circuits that are explained thoroughly in the following sections.

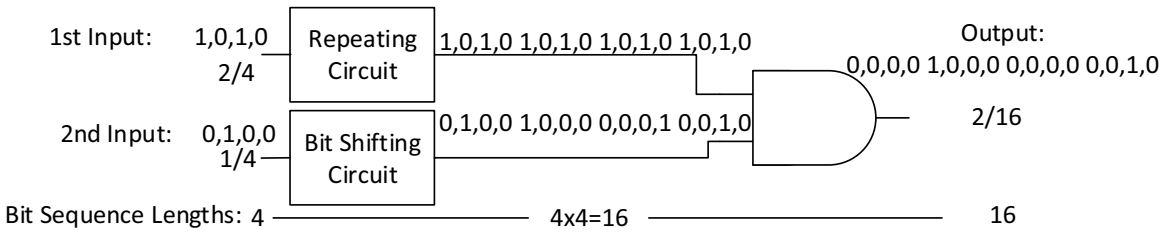


Fig. 4. A block level demonstration of the proposed 2-input stochastic multiplication circuit.

1) Repeating Circuit:

Consider input bit sequences consisting of *n* bits. The repeating circuit repeats one of the input sequences *n* times. This is illustrated in Fig. 4. A circuit exhibiting this behavior can be designed with a multiplexer and a delay block as shown in Fig. 5. The multiplexer's inputs are the input of the repeating circuit and its delayed version. It determines to pass whether the input sequence or the delayed repeated sequence using SEL-0. At first, the input sequence is directly passed to the output. Then the output signal is delayed and returns to the output using the select input of the multiplexer. The second process is repeated *n*-1 times. The delay block makes the time alignment of the repeated sequence with the output. Note that delay block makes a delay time as (bit-width × *n*).

To keep the scalability advantage of the proposed circuit, the delay block should be able to change its delay time for different input bit sizes, without any change in its structure. Tunable delay blocks, being able to digitally adjust the amount of delay, perfectly suits for this purpose [20, 21]. Transistor-level representation of the used delay block is shown in Fig. 6. The block offers delay values varying from a unit delay, 1X delay, by applying 0s to all 3 DIG inputs, to 8X delay by applying 1s to all 3 DIG inputs where X is the width of a single bit. Normally and expectedly, the original delay circuit cannot successfully give delays that are considerably larger than X. To achieve a delay as large as 8X, we add 5 extra inverters to the original delay circuit that initially has an input stage cascaded with two inverters at the output. An added inverter is shown inside dashed lines in Fig. 6.

In order to elucidate the operation of the repeating circuit, we give an explicit example with signal forms in Fig. 7.

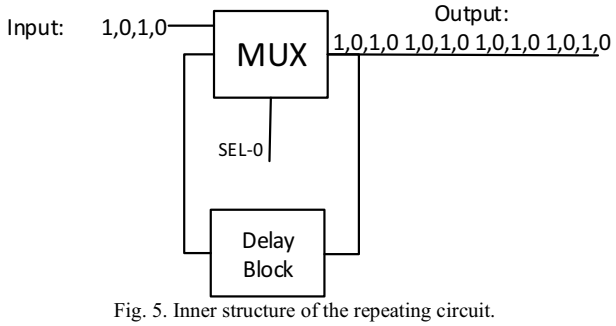


Fig. 5. Inner structure of the repeating circuit.

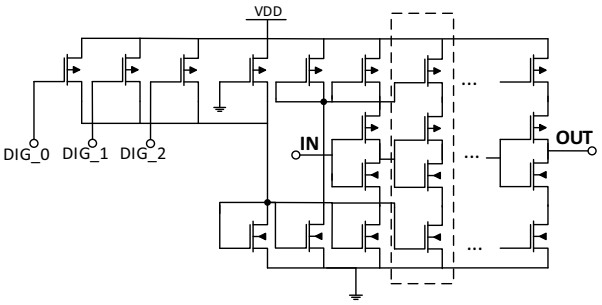


Fig. 6. Tunable delay block structure.

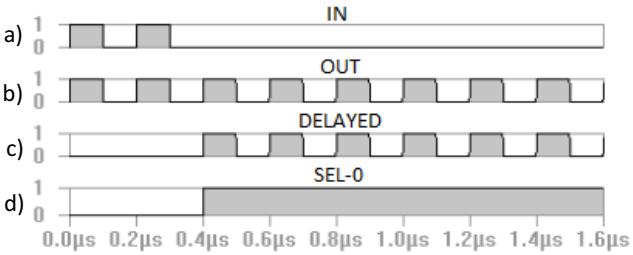


Fig. 7. Signal forms of the repeating circuit: a) input (1,0,1,0 - each bit having 100ns bit width), b) output, c) delayed 2nd input of the MUX, d) SEL-0 (Select input of the MUX).

2) Bit Shifting Circuit:

Consider input bit sequences consisting of *n* bits. The bit shifting circuit uses the input bit sequence and performs one-bit shift operation for *n*-1 times. This is illustrated in Fig. 4. We design a circuit exhibiting this behavior with two multiplexers, a D-flip-flop, and a delay block as shown in Fig. 8. This circuit fundamentally works as keeping the first bit of the input sequence, assigning it to the last bit position, and repeating this for *n* times.

In Fig. 8, MUX-1 works in the same manner as the multiplexer in the repeating circuit. MUX-2 is used for placing the first bit, which is kept by the D-flip-flop, to the last position in the shifting operation. This placing process is repeated *n*-1 times. Here, the D-flip-flop is used for keeping the first bit of the each shifted sequence until all other bits are passed. Note that the delay block makes a delay time as (bit-width × *n*-1).

In Fig. 9, transient analysis results of the bit shifting circuit for a 4-bit input 1,0,0,0 - each bit having 100ns bit width - are given. Fig. 9a) and Fig. 9.b) correspond to the input and the output, respectively. Fig. 9c) and Fig. 9e) show select inputs of MUX-1 and MUX-2, respectively. Fig. 9d) corresponds to the output of the D-flip-flop. Fig. 9f) is the

shifted version of the last *n*-bit output sequence from the output of MUX-2. Fig. 9g), corresponding to one of the inputs of MUX-1, is the shifted and the delayed version of the last *n*-bit output sequence.

Fig. 10 shows the whole multiplication circuit. This circuit can be used for different input bit counts with minor modifications of select inputs. We show that the select inputs can be generated for different bit counts by using a clock signal having 50% duty cycle, D-flip-flops, and OR-gates. The corresponding circuit and the clock input signal are given in Fig. 11 and Fig. 12, respectively. Here, D-flip-flops are used for frequency division of clock signals and OR gates are used to merge the signals. In Fig. 11, the generation of the select inputs for 4-bit and 8-bit input sequences are explicitly shown; the signal forms are given in Fig. 13.

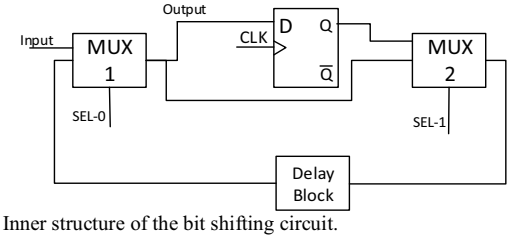


Fig. 8. Inner structure of the bit shifting circuit.

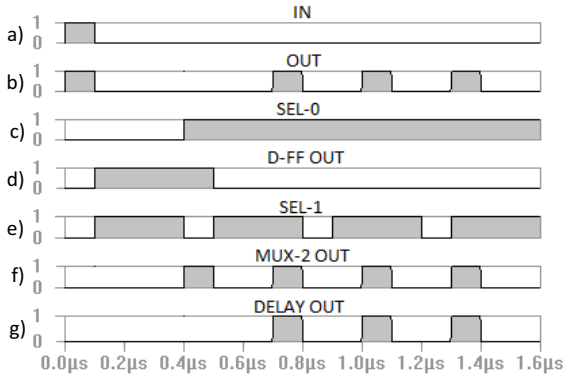


Fig. 9. Signal variances of the nodes for the bit shifting circuit; each bit having a 100ns bit width.

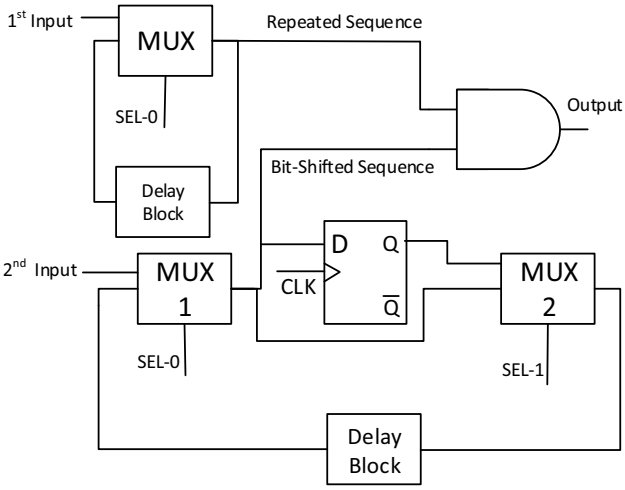


Fig. 10. Proposed stochastic multiplication circuit.

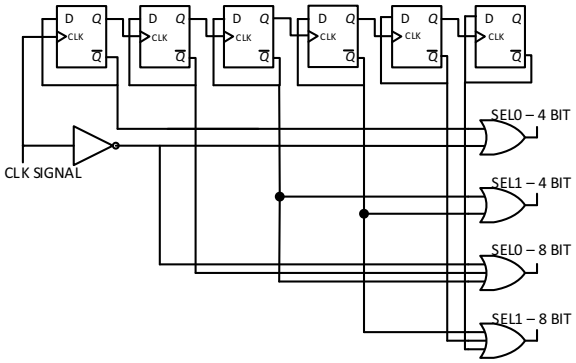


Fig. 11. Select signal generation circuit.

Fig. 12. Input CLK signal of Fig. 11.

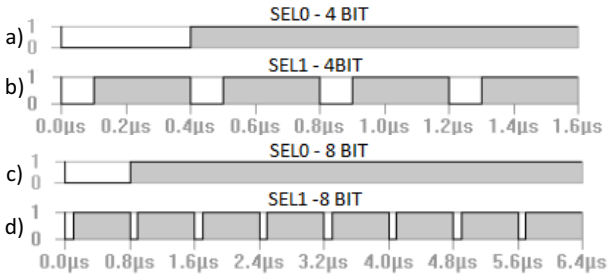


Fig. 13. Generated select inputs.

B. *Scaled Addition*

Due to an obligation that all values should lie in an [0, 1] interval in stochastic computing, addition is performed using a scaled addition process which gives an arithmetic average of inputs. For example, consider input bit streams X_1=3/5 and X_2=4/5. Stochastic addition of these inputs results in Y=(3/5+4/5)/2=7/10. Two-input stochastic addition is conventionally implemented using a 2-to-1 MUX with applying a 0.5 valued random bit stream to its select input. As for any block performing conventional stochastic computing, the output bit stream of the MUX has a Binomial distribution, previously stated in Theorem 1. This results in a similar accuracy problem as we face for an AND gate previously illustrated in Fig. 3.

In order to achieve accurate stochastic addition we show a necessary condition with the following lemma.

Lemma-2: *Consider m input bit streams, each having n bits with taking values between 1/n and n/n. Accurate stochastic addition of these inputs requires an output bit stream having at least (m×n) bits.*

Proof: *Consider the worst case scenario for which each input takes the value of 1/n. The output value should be 1/(m×n) that requires (m×n) bits.* □

Lemma-2 tells us that to perform error-free addition of two *n*-bit inputs, we optimally need a 2*n*-bit output. We satisfy this with circuit in Fig. 14.

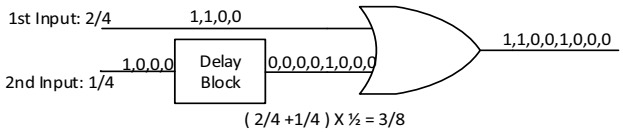


Fig. 14. Proposed 2-input stochastic scaled addition circuit.

The delay block postpones one of the inputs with the time length of n-bits, so we can separately process the input bit streams. For the delay block we use the same circuit that we use for the multiplication in the previous section. Recall that the delay block is scalable that can change its delay time for different bit sizes by applying different input assignments of 0s and 1s.

III. Experimental Results

We evaluate the proposed circuits in terms of area, delay, and accuracy. Simulations are performed in Spice by using 0.18um CMOS technology. To implement the circuit blocks, we use NAND2 gates with delay values derived from Spice simulations. The only exception is the delay block that has its own transistor level circuit. For delay calculations, we first determine the minimum bit width needed for 1-bit operation, by considering gates delays. We use 0.125 ns for each bit length. Then the total delay is calculated by multiplying this value with the number of bits in the output stream with an addition of interior gate delays. Interior delays constitute negligibly small portion of the total delay, especially for long output bit sequences. For area calculations, we calculate the number of transistors. In terms of accuracy, the proposed circuits are always %100 accurate.

Table 1 and 2 present area-delay-accuracy performance of the proposed circuits in comparison with the conventional stochastic multiplication circuit (AND gate consisting of 2 NAND gates) and the scaled addition circuit (2-to-1 MUX consisting of 4 NAND gates), respectively. In each circuit, for given input probabilities (levels/values) and minimum accuracy conditions, minimum necessary number of bits of input bit sequences are found. Then, corresponding delay values are obtained. Accuracy values of the conventional stochastic circuits are evaluated using the average relative error rates extracted from a Monte Carlo method in which the input bit sequences are randomly generated ($Accuracy = 1 - rel.error\ rate$). Accuracy values are obtained using the worst case accuracy conditions for different input/output values. For our circuits, 100% accuracy or 0% error is always guaranteed that is independent of the input/output values. However, for a conventional stochastic circuit, the relative error rate $\sqrt{\frac{(1-p)}{pn}}$ directly depends on the expected output value p (n is the number of bits in the output stream). Therefore, to obtain accuracy values for the worst case, minimum output values should be considered; correspondingly minimum input levels/values of 1/4 and 1/8 are selected.

Results in the tables indicate that the proposed circuits always overwhelm the conventional ones in terms of accuracy, delay, and area-delay product. This is so dramatic for cases having 99% or higher accuracy. If input probabilities of 1/8 and accuracy values of 99.9% are selected then the area-delay products of the proposed circuits are approximately 15000 and 45000 times better compared to those of the conventional ones for multiplication and scaled addition operations, respectively.

For further evaluation of our work, we can also consider studies in the literature aiming for improving accuracy in stochastic computing, but their area-delay performance results are far beyond the results of the proposed work. Accurate multiplication has been obtained earlier by extremely increasing the area [7]. There are also other works claiming 0% mean error [9, 17], however the term "mean error" does not mean 0% deviation from the expected value, so the accuracy performances, according to our definitions, are very poor. On the other hand, the proposed circuits have 0% mean error as well as 0% standard deviation.

Another evaluation can be done by comparing the proposed circuits with their deterministic counterparts. Fig. 15 shows a comparison for 4-bit and 8-bit multiplication operations. The proposed stochastic method generally has a better area performance compared to the deterministic binary ones. This is much more significant for 8-bit operations (over 20 times smaller area). Note that the difference would increase for higher bit operations. Of course, a rigorous comparison would depend on the specifics of the types of technology used. Additionally, other performance metrics should be considered such as delay, power, sensitivity, and scalability. Nevertheless, these comparisons certainly show the potential of the proposed circuits.

TABLE 1. Performance Comparison Of The Conventional Stochastic Multiplier And The Proposed One.

	Input Levels: 4 (1/4, 2/4, …, 4/4) Selected Input Levels/Values: 1/4 Expected Output Value: 1/16				**Input Levels: 8 (1/8, 2/8, ..., 8/8) Selected Input Levels/Values: 1/8 Expected Output Value: 1/64**			
Circuit	**Conventional**			**Proposed**	**Conventional**			**Proposed**
Area (Transistor Count - T.C.)	8	8	8	**146**	8	8	8	**146**
Accuracy (%)	90	99	99.9	**100**	90	99	99.9	**100**
Number of Input Bits	512	32768	4194304	**4**	2048	262144	16777216	**8**
Delay (ns)	64.13	4096.1	524288.1	**4.225**	256.13	32768.125	2097152.13	**8.225**
Area × Delay	513	32769	4194305	**616.85**	2049.04	262145	16777217.04	**1200.85**

TABLE 2. PERFORMANCE COMPARISON OF THE CONVENTIONAL STOCHASTIC ADDER AND THE PROPOSED ONE.

	Input Levels: 4 (1/4, 2/4, …, 4/4) Selected Input Levels/Values:1/4 Expected Output Probability: 1/4				Input Levels: 8 (1/8, 2/8, …, 8/8) Selected Input Levels/Values: 1/8 Expected Output Probability: 1/8			
Circuit	**Conventional**			**Proposed**	**Conventional**			**Proposed**
Area (Transistor Count - T.C.)	16	16	16	**45**	16	16	16	**45**
Accuracy (%)	90	99	99.9	**100**	90	99	99.9	**100**
Number of Input Bits	128	8192	1048576	**4**	256	16384	2097152	**8**
Delay (ns)	16.13	1024.1	131072.1	**1.125**	32.125	2048.125	262144.13	**2.125**
Area × Delay	258	16386	2097154	**50.625**	514	32700	4194306.08	**95.625**

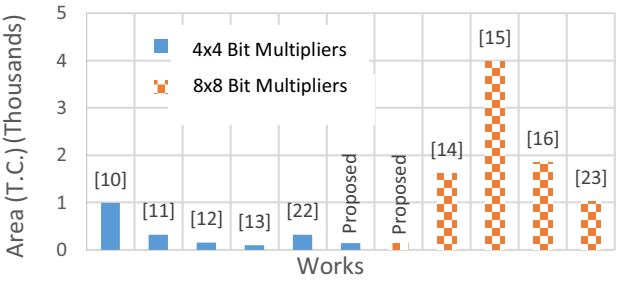


Fig. 15. Area (transistor counts) comparison of deterministic multipliers and the proposed one.

IV. CONCLUSION

This work presents a method curing the accuracy problem (or astronomic delay) of the stochastic computing, which is the most significant drawback of it. Error-free multiplication and addition circuits are proposed by exploiting correlation in stochastic input bit sequences. The proposed circuits are the first in the literature offering scalable and 100% accurate stochastic arithmetic operations. As a future work, we will develop a performance optimization algorithm for the proposed methodology with a detailed transistor-level timing analysis, so we will be able to effectively implement complex circuit blocks including processors and state machines.

REFERENCES

[1] J. von Neumann, 1963, "Probabilistic logics and the synthesis of reliable organisms from unreliable components", The Collected Works of John von Neumann, Macmillan.

[2] B.R. Gaines, 1967, "Stochastic computing", Proc. AFIPS Spring Joint Computer Conf., 149-156.

[3] A. Alaghi and J. P. Hayes, 2013, "Survey of stochastic computing", ACM Trans. Embedded Computing Systems, 12 (2s), 1-16.

[4] B. Zelkin, 2004. U.S. Patent No. 6,745,219. Washington, DC: U.S. Patent and Trademark Office.

[5] A. Alaghi and J. P. Hayes, 2014, "Fast and accurate computation using stochastic circuits", Automation and Test in Europe Conference and Exhibition, 1-4.

[6] W.S. Wong and A. Salleo, (Eds.). 2009. Flexible electronics: materials and applications (Vol. 11). Springer Science & Business Media.

[7] P. K. Gupta and R. Kumaresan, 1988, "Binary multiplication with PN sequences", IEEE Trans. Acoustics. Speech, and Signal Process., 36, 603-606.

[8] W. Qian, X. Li, M. Riedel, K. Bazargan, and D. Lilja, 2011 "An architecture for fault-tolerant computation with stochastic logic," IEEE Trans. on Computers, vol.60, no.1, pp.93-105.

[9] A. Alaghi and J. P. Hayes, 2013, October. Exploiting correlation in stochastic circuit design. In Computer Design (ICCD), 2013 IEEE 31st Int. Conference on (pp. 39-46).

[10] N. A. Nayan, Y. Takahashi, and Sekine, T. (2012). LSI implementation of a low-power 4× 4-bit array two-phase clocked adiabatic static CMOS logic multiplier. Microelectronics Journal, 43(4), 244-249.

[11] N. Ravi, A. Satish, T.J. Prasad, and T.S. Rao, "A new design for array multiplier with trade off in power and area", arXiv preprint arXiv: 1111.7258, 2011.

[12] V.K. Rongali and B. Srinivas, "Design of area efficient high speed parallel multiplier using low power technique on 0.18um technology", Int. Journal of Advanced Research in Computer Eng. & Technology (IJARCET) Vol. 2, 2013.

[13] M. Prakash, "Area efficient parallel multipliers using pass transistor logic (PTL)", International Journal of Innovative Research in Science, Engineering and Tech., Vol. 4, 2015.

[14] N. Ravi, T.S. Rao, and T.J. Prasad, "Pipelined C2 MOS register high speed modified Booth multiplier", Int. Journal of Advanced Networking and Applications, 1031-1034, 2011.

[15] J.T. Yan and Z.W. Chen, "Low-power multiplier design with row and column bypassing", SOC. IEEE International Conference, 2009. SOCC 2009.

[16] S. Agarwal, V. K. Pavankumar and R. Yokesh, (2008, January). Energy-efficient, high performance circuits for arithmetic units. In VLSI Design, 2008. VLSID 2008. 21st International Conference on (pp. 371-376). IEEE.

[17] A. Alaghi and J. P. Hayes, 2015, May. On the functions realized by stochastic computing circuits. In Proceedings of the 25th Great Lakes Symp. on VLSI (pp. 331-336). ACM.

[18] H. Ichihara, S. Ishii, D. Sunamori, T. Iwagaki, and T. Inoue, 2014, October. Compact and accurate stochastic circuits with shared random number sources. In Computer Design (ICCD), 2014 32nd IEEE International Conference on (pp. 361-366).

[19] K. Ishida, et al. (2014). Improvement and Applications of Large-Area Flexible Electronics with Organic Transistors. In T. Wojcicki (Ed.), VLSI: Circuits for Emerging Applications (pp. 95-110). CRC Press.

[20] M. Maymandi-Nejad and M. Sachdev (2005). A monotonic digitally controlled delay element. Solid-State Circuits, IEEE Journal of, 40(11), 2212-2219.

[21] S. B. Kobenge and H. Yang, (2009, May). Power optimized digitally programmable delay element. In S. Chen (Ed.), WSEAS International Conference. Proceedings. Mathematics and Computers in Science and Engineering (No. 9). WSEAS.

[22] J. Gupta, A. Grover, G. K. Wadhwa, and N. Grover (2013, August). Multipliers using low power adder cells using 180nm Technology. In Computational and Business Intelligence (ISCBI), 2013 International Symposium on (pp. 3-6). IEEE.

[23] D. R. Gandhi and N. N. Shah, (2013, March). Comparative analysis for hardware circuit architecture of Wallace tree multiplier. In Intelligent Systems and Signal Processing (ISSP), 2013 International Conference on (pp. 1-6). IEEE.