Enhancing Stochastic Computations via Process Variation

Rui Policarpo Duarte
Departamento de Ciências e Tecnologias
Universidade Autónoma de Lisboa, Portugal
Email: rpduarte@ual.pt

Mario Véstias and Horácio Neto
INESC-ID, Instituto Superior Técnico,
Universidade Técnica de Lisboa, Portugal
Email: mvestias@inesc-id.pt, hcn@inesc-id.pt

***Abstract*—Stochastic computing has emerged as a computational paradigm that offers arithmetic operators with high-performance, compact implementations and robust to errors by producing approximate results. This work addresses two of the major limitations for its implementation which affects its accuracy: the correlation between stochastic bitstreams and the unobserved signal transitions. A novel implementation of stochastic arithmetic building-blocks is proposed to improve the quality of the results. It relies on Self-Timed Ring-Oscillators to produce different clock signals with different clock frequencies, by taking advantage of the influence of process variation in the timing of the logic elements on the FPGA. This work also presents an automated test platform for stochastic systems, which was used to evaluate the impact of the proposed enhancements. Tests were performed to compare both proposed and typical implementations, on reconfigurable devices with 28 nm and 60 nm fabrication processes. Finally, presented results demonstrate that the proposed architectures subjected to the impact of process variation improve the quality of the results.**

I. Introduction

As the fabrication process scales down, devices are manufactured with increased variability. This is known as process variation and occurs because transistors are not manufactured with the same physical characteristics, thus making devices to perform unevenly. An example of a countermeasure to compensate for this variation such as variation-aware placement methods for Field-Programmable Gate Arrays (FPGAs) [1], here each FPGA is pre-characterised and its variation map classified according to the median of the variation pattern, to perform variation-aware placement of high-performance deterministic logical systems.

On the other hand, stochastic computing [2] has emerged as a computational paradigm which offers, low complexity implementation of arithmetic operators, high resilience to errors, and high performance [3]. Besides DSP applications, examples of applications include neuromorphic and bio-inspired systems: binary synapses for low-power neuromorphic systems [4], and digital neurosynaptic network for neuromorphic chips to develop brain-like computational structures [5].

Event though they are more resilient to errors, stochastic system are synchronous, which means that their behaviour happens at regular occasions in time, even when operating under timing or soft-errors. Although this may seem desirable, it constitutes the ground for one of the limitations of stochastic computing: when signals become correlated, either through weak stochastic generation, or due to concatenated arithmetic operations, the values in the resulting bitstreams tend to be incorrect.

The proposed improvement for this problem, is to clock the registers in the stochastic arithmetic units through spread-spectrum clocks, to register the signals at different times and reduce the effects of signal correlation and synchronisation. Notwithstanding the presence of Phase-Locked Loops (PLLs) with support for spread-spectrum in modern FPGA devices, there aren't enough available to implement real-life stochastic systems.

Forasmuch as process variation affects intra-die and inter-die transistors, in a device family, clock generators, also known as Self-Timed Ring-Oscillator (STRO), created at the expense of Logic Elements (LEs) will have their clock signals affected by process variation.

Moreover, process variation affects the performance of transistors on the device. Their performance is also affected by variation of its voltage and temperature. Hence, the new clock signals driving the stochastic arithmetic units will have additional entropy derived from changes in voltage and temperature. Furthermore, by avoiding the use of the global clock tree, the proposed stochastic arithmetic blocks, uses less power.

In consideration of the tendency for process variation to increase, it is of interest to explore it via architectures and methods to improve the performance and accuracy of stochastic systems.

Since currently it is not possible to simulate exactly the behaviour of the STROs taking into account the impact of process variation, therefore a test platform was created to implement and run the stochastic circuits on FPGAs. FPGAs were adopted because as semiconductor devices they suffer from process variation; also because of the their reconfigurability capabilities, which allows to test the same circuit on different locations of the device, and with different parameters; and they offer bit-level circuit specification. At the moment there's no other alternative technology which supports these features.

The main contributions of this work are:

- study on the impact of process variation on the accuracy of stochastic units;
- enhanced units for stochastic computations using spread-spectrum clocks;
- lightweight, but generic and scalable test platform for stochastic circuits and systems;

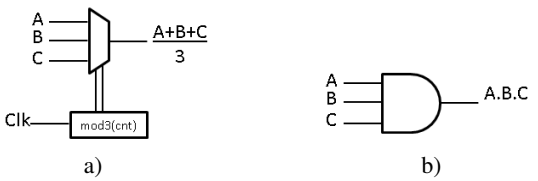

Fig. 1. Block diagram of the a) stochastic adder and b) multiplier with 3 inputs.

- benefits and flexibility on the implementation of stochastic bitstreams: tradeoff speed and power and accuracy, when compared to typical implementations.

II. Background

A. Stochastic Computing

The concept of designing probabilistic logics and synthesis of robust systems from unreliable components has been presented in the past, by [6] and by [2] as an alternative number representation scheme which provides high tolerance to errors and compact operators than conventional representation schemes, e.g. fixed and floating-point binary representations.

Examples of the applicability of stochastic computing can be found in different classes of problems such as Finite Impulse Response (FIR) [7], and Infinite Impulse Response (IIR) [8] digital filters, neural network controller for small wind turbine systems [9], decoding of Low-Density Parity Code (LDPC) codes [10], [11], and high-throughput Bayesian computing machines [12]. In [13], an FPGA implementation of a probabilistic neural network is proposed and developed to decode motor cortical ensemble recordings in rats performing a lever-pressing task for water rewards.

Essentially, stochastic signals are defined as generated by a memoryless continuous-time stochastic process producing two distinct values. A stochastic bitstream, according to [14], is defined as a sequence of stochastic signals over time, where its value is defined as the number of ones over the total number of bits. Stochastic numbers are represented as probabilities, and thus, they are in the interval $[0, 1]$. This contribution also introduces details about hardware realisations of the building-blocks for stochastic systems.

Basic stochastic arithmetic units and other complex stochastic arithmetic operators are presented in [15] where the computational elements are employed in artificial neural networks. Figure 1 shows the block diagram, of 3-input stochastic a) adder and b) multiplier. The stochastic multiplication corresponds to the logic AND of all stochastic inputs, while addition is obtained via a round-robin multiplexation of the stochastic inputs, which is implemented with a N-module counter corresponding to N inputs in the multiplexer.

To perform complex arithmetic computations on stochastic bitstreams, [16] has proposed a methodology to generate Finite State Machine (FSM) topologies. [17] applies stochastic computations to Markov-Chain Monte Carlo algorithms for inference applications. A survey covering most of the work done on the stochastic arithmetic units has been presented in [18].

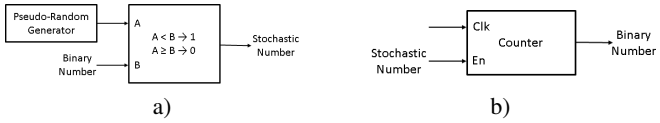

Fig. 2. Block diagram of a) binary-to-stochastic and b) stochastic-to-binary units.

The fact that many DSP applications for image and video processing are based on addition and multiplication, and are tolerant to some errors in their computations, makes stochastic computing appealing to implement. On this account, [3] applied the concept of stochastic logic to a reconfigurable architecture that implements processing operations on a datapath. Analysis was performed on cost and on the sources of error: approximation, quantisation, and random fluctuations, and found that stochastic computing is more tolerant of soft errors, or bit flips, than the deterministic implementations. It was also found that the quality of the results degrades gracefully with the increase of errors.

The main drawbacks in the implementation of stochastic computing are: the linear increase in the precision of stochastic computations requires an exponential increase in the length of the bitstream; sensitivity to temporal correlations; the supporting blocks are usually the performance bottleneck, instead of the arithmetic units.

One of such supporting blocks is the binary-to-stochastic conversion, presented in Figure 2 a). This block is constituted of a pseudo-random number generator, of which the output is compared to be less or equal than a binary quantity. Over time, the numbers in that condition represent a ratio between 0 and 1. [19] presented a method for generation of probabilities derived from combinatorial operations on other stochastic values. [20] proposed the generation of stochastic bitstreams using generators based on FSM that emulate reversible Markov-Chains. The conversion from stochastic-to-binary is based on the integration of the 1s on a bitstream, which is accomplished using a binary counter. It is illustrated in Figure 2 b).

B. Process Variation

Variation in the sizes of the physical structures of transistors affects their electrical characteristics, across the device, namely threshold voltage, carrier mobility, impedances and current leakages, and consequently propagation delay and power consumption. This type of variation, also known as process variation, can be observed on FPGAs through a characterisation of the device by measuring the delay for each basic element on the device. Various contributions [21], [22], [23] present different methods and measurements made for many devices from the same family. Moreover, silicon devices are sensitive to aging, changes in supply voltage, temperature, signal cross talk and jitter [24]. Furthermore, designs targeting implementation on FPGAs endure extra variation from placement and routing.

The aforementioned variation sources affect the delay of paths in a circuit, thus leading to degradation of the maximum clock frequency. Table I summarises the most noticeable types of variation and how they influence the delay of the paths in the circuit.

Type of Variation	Source	Path-Delay Increases With
Process variation	Fabrication	Process size reduction
Voltage	Low-power circuits	Power decrease
Temperature	Environment	Temperature increase[1]
Jitter	Fabrication	Placement and routing
Degradation	Aging	Time

TABLE I. SUMMARY OF THE DIFFERENT TYPES OF VARIATION, THEIR ORIGIN AND THEIR CONTRIBUTION FOR THE DELAY INCREASE IN CIRCUIT PATHS.

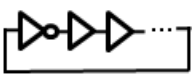

Fig. 3. Block diagram of a self-timed ring oscillator.

So far, process variation is considered undesirable and has been addressed as a problem that needs to be mitigated. Nevertheless, Physical Unclonable Functions (PUFs) try to use it to extract (unique) information about the device, in order to identify it [25]. It uses STROs exploit the inherent features of random process variations by producing varying clock frequencies. These unpredictable variations in frequencies are captured and used to identify the device and prevent circuit replicas to operate correctly, or disable them.

Identically, the work presented in this paper tries to take advantage of process variation to improve the statistical independence of the stochastic bitstreams, and thus the accuracy of the stochastic computations. The block diagram of a ring-oscillator is represented in figure 3. The inverter gate is connected to a chain of pass-through, or delay, gates; and their output to the input of the inverter gate. This makes it toggle the value of the inverter after it is propagated through the gates. The clock frequency of this oscillator is determined by the number of gates in the loop. The duty cycle is 50%.

III. CONCEPT AND PROPOSED STOCHASTIC UNITS

If two stochastic bitstreams become correlated, either due to a weak implementation of the Bin2Sto block, or consequence of a sequence of arithmetic operations, they may result in an incorrect result. As an example, multiplication (AND) of two numbers with 0.5 value, and a bitstreams with values 11001100 and 00110011, will result in a bitstream with 00000000, which is not the expected result (0.25). If any of the signals were to be time shifted, the result could still be incorrect, but it would be *closer* to the correct value.

The main idea behind the work presented is to deviate from synchronous deterministic stochastic circuit designs, to minimise *unobserved* signal transitions in stochastic bitstreams due to clock synchronism. *Unobserved* stochastic signal transitions correspond to the portions of the bitstream that don't contribute to the final result. They can occur because: time multiplexing of stochastic bitstreams, and different clocks in the processing elements of a stochastic bitstream. In that direction, this work proposes to use delay gates and STRO circuits to introduce differences in the frequency and phase of the clocks for each clocked stochastic unit. Moreover, by having their timing dependent on process, voltage, temperature and aging of the device, the ring-oscillators will constantly change their clock frequencies, thus increasing the clock's entropy and reducing signal correlation, in the stochastic bitstreams, minimising the synchronism between them.

Another benefit is the reduction in the power dissipated in the clock trees, without the drawback of having to implement synchronisers and other structures usually found in asynchronous circuit designs [26]. The main cost of implementing this concept is the extra Logic Elements consumed by the STROs and delay elements. Recent FPGAs have PLLs that support spread spectrum, but their number on the largest device is still very limited, below 32, when compared to the number of stochastic units requiring by a useful system.

A. Self-Timed Ring Oscillator

This is the unit that generates the clock signal for all stochastic units, and replaces the existing clock sources in them. The strength of using a STRO instead of common clock sources, such as PLLs, are:

- Availability: it is possible to assign a different one for each arithmetic unit;
- Variability: all signals generated by a STRO will vary its period with the variation of voltage, temperature, location on the device and its degradation;
- Flexibility: by adopting different STRO architectures, to generate different clock frequencies, duty-cycles, or even other signal patterns for specific applications, i.e. research on accuracy and power optimisation.

Depending on the requirements of the design, the STRO can be parameterised accordingly. A low frequency STRO, allows to reduce the number of signal transitions, and hence, reducing the consumed power, but it requires more logic elements in the loop of the STRO, and takes longer to converge to the result. On the other hand, a high frequency STRO converges to the result faster, requires less hardware resources, but it will consume more power. The circuit resources, or LEs, occupied by the STRO are directly proportional to the number of elements in the loop.

B. Self-Timed Binary-to-Stochastic and Stochastic-to-Binary Converters

As aforementioned, the binary-to-stochastic converter is based on a pseudo-random generator, usually implemented using an Linear Feedback Shift-Register (LFSR) [27], whose output is compared with the binary value being encoded in a stochastic bitstream [18]. Other contributions such as [19] and [20] have presented methods to produce stochastic bitstreams with specific values at the expense of a few stochastic bitstreams with constant values.

In this work, the main interest is to add a STRO to the the input of an LFSR, instead of the system clock. Besides the seed inside each LFSR, the new stochastic bitstream generator the circuit now has parameters such as clock frequency and phase to further reduce the correlation between bitstreams.

In a stochastic system, the stochastic-to-binary converter is the only unit that can be connected to the external clock. This

[1]In fabrication processes below 65 nm the opposite effect can be observed as a consequence of temperature inversion. Source: http://tech.tdzire.com/what-is-temperature-inversion/

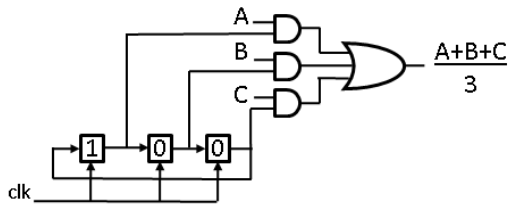

Fig. 4. RTL of the stochastic adder with 3 inputs.

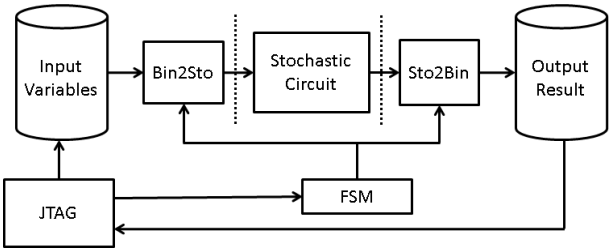

Fig. 5. Block diagram of the test circuit for the stochastic units.

is possible since no other clocked component in the system uses that signal. Nevertheless, a STRO can generate the input clock for that unit as well.

C. Self-Timed Adder

Addition is one of the basic arithmetic building-blocks. In stochastic computing the adder is comprised of a round-robin input selector that passes its value to the output. The result corresponds to the sum of values from all input bitstreams averaged by their number.

The proposed solution uses a STRO to clock the input selector register and it optimises typical implementation by adopting the one-hot encoding for the selection of the input, hence avoiding to implement and N-module binary adder and decoder. This implementation only changes 2 registers per clock, which is translated in power savings when compared to the typical implementation. A 3-input adder circuit is illustrated in figure 4.

Furthermore, the stochastic adder can consider different weights on its inputs. This can be accomplished by assigning more than one bit for an input, corresponding to the weight. In this case, the result at the output is scaled down by the total number of weights assigned.

IV. Test Platform

A lightweight, but scalable, test platform was created to test the stochastic arithmetic units and systems, and asses the impact of the proposed architectures. This test platform generates all the stimulus signals, as stochastic bitstreams, required by the stochastic circuit under test. It produces conversion of the resulting stochastic bitstreams to be read by the host computer.

The main features of the test platform are:

- Scalability: supports any number of inputs, outputs and stochastic circuits under test simultaneously.
- Parameterisable: supports the specification of the length of the stochastic bitstreams and parameters of the stochastic units;
- Interface with host: data transfers between the host computer and the stochastic circuit under test;
- Automated: stand-alone process to run and analyse the tests results.

This test platform is described in VHDL, and is synthesised to configure EP4CE22F17C6 Cyclone IV [28] and 5CEFA2F23C8 Cyclone V [29] FPGAs, from Altera. Nevertheless, it can be easily adapted to any other reconfigurable platforms.

A. Circuit Architecture

The circuit of the test platform is constituted by a generator of the stochastic input stimulus, the stochastic circuit under test, and the stochastic output converter. Figure 5 depicts the block diagram of the test circuit with one instance of a circuit under test.

On the left of the figure, there is the memory storage (Block Random Access Mememorys (BRAMs)) for the input values, followed by the binary-to-stochastic converter block (Bin2Sto), which generates the stochastic bitstreams. On the right of the figure, there is the stochastic-to-binary converter block (Sto2Bin), which is a sink that captures the stochastic bitstreams and converts it to a binary value, along with the memory storage for the binary values (BRAMs). The stochastic circuit under test performs the computations over the set of input values, is located in the centre of the figure and sandwiched between the stochastic generator and the stochastic sink. The FSM, in the bottom, controls the test execution and provides all the control signals required in the design. Moreover, this unit receives the trigger signal, via Joint Test Action Group (JTAG), to initiate the test. The FSM also sets the seed values in the binary-to-stochastic units, and resets the integrator in the stochastic-to-binary units. This circuit also supports many stochastic units to be tested in parallel, as the Bin2Sto and the Sto2Bin blocks support many stochastic bitstreams simultaneously.

The input data is stored in BRAMs and consists of the values for the input variables, encoded in unsigned binary. The output of the stochastic unit is represented as the sum of the number of 1s over the total number of bits in the stochastic bitstream.

B. Operation

Once the FPGA is configured with the test circuit, it is ready to exchange data with the host computer and start the test. The test platform offers the following commands to be sent from the host computer: 1. load input values (binary); 2. set design parameters; 3. start the test; 4. check if the test has concluded; 5. retrieve the results from the FPGA (binary).

The circuit operation has been automated to access and edit the contents of the BRAMs on the test circuit via the host computer. Once the trigger signal is sent to the FSM, it starts the execution of the test, and sets a BUSY flag. When the test completes, the FSM signals it by clearing the BUSY flag, so the host computer acknowledges the end of the test and retrieves the results from the FPGA.

Clk	Mean Error	Variance
STRO	2.295E-4	4.609E-5
Ext	6.2E-3	6.673E-5

TABLE II. ABSOLUTE ERROR METRICS

C. Scalability

It is of great interest to have a generic description of the design which supports problems of any size. Thus, all input and output ports in the stochastic circuit under test are grouped as buses, whose length changes according to the number of inputs and outputs, and the number of stochastic circuits being tested.

V. EXPERIMENTAL RESULTS

To assess the impact of the enhancements on the proposed stochastic units, their performance was compared contra the performance of typical implementations of the same stochastic units using an external clock source. In the present case, it was considered a 3-input adder, connected to 3 binary-to-stochastic converters and a stochastic-to-binary converter. All tests were performed, using Cyclones IV (60nm) and V (28nm) FPGAs, at room temperature, over stochastic bitstream with 4096 bits and repeated 100 times.

A. *External Clock* vs *Ring-Oscillator*

To faithfully compare the two circuits using different clock sources, with similar clock frequencies. The external clock was set to 50 MHz and the ring-oscillator to 54 MHz. The clock frequency of the second was set by adjusting the number of logic elements in it, and measuring in an output pin. These circuits were placed on the same locations (X18,Y9) on the device, and had the same test conditions, i.e. length of the stochastic bitstream, pseudo-random seeds, and input values.

The absolute error is defined as the difference between the expected result and the actual result read from the FPGA. Figure 6 shows the absolute error and Figure 7 shows its profile. Table II shows the mean and variance of the error for both tests.

From these results, it is observable that the circuit using the STRO has greater accuracy than the typical implementation, as their data point are closer to zero. Other experiments shown that it is possible to obtain more accurate results while operating a at a lower clock frequency, hence consuming less power.

B. Intra-Die Process Variation

To assess the impact of process variation on the results, two instances of the same stochastic adder were placed on different locations, Loc1 @(X20,Y23) and Loc2 @(X34,Y23), on a Cyclone V FPGA. Both units used the same stochastic bitstreams as inputs and had the STRO running at 133.6 MHz.

Figure 8 shows that stochastic adders in different locations produce different result values, where the 0.5 value was expected. Loc1 proved to be more accurate than Loc2. Compared to the previous test it is observable that with the increase of the clock frequency, the accuracy of the stochastic unit has increased.

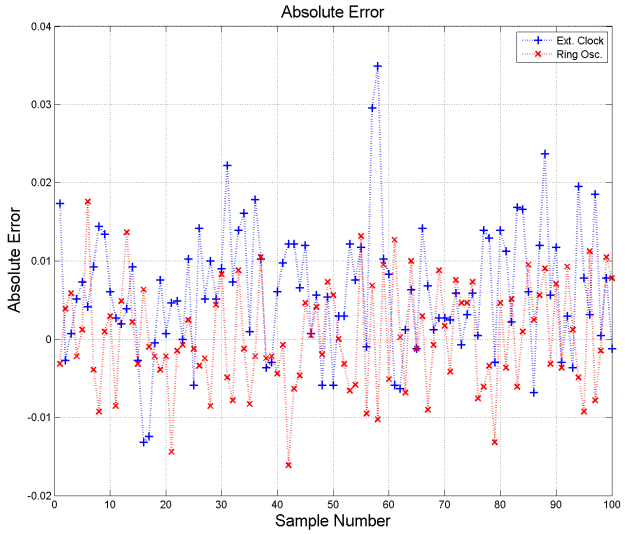

Fig. 6. Comparison between stochastic adders using system clock and a local ring-oscillator.

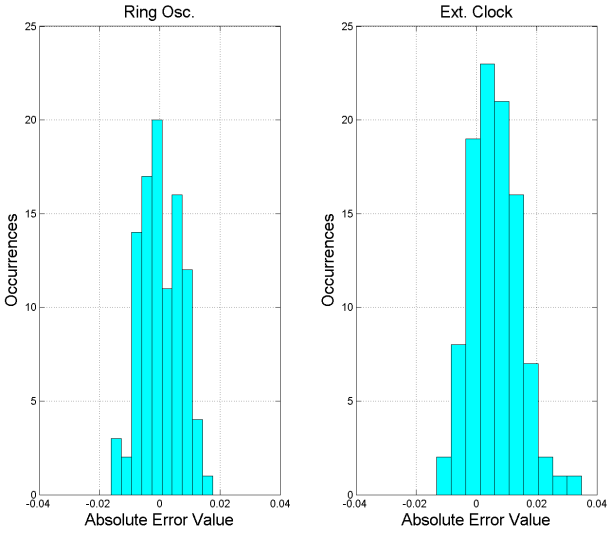

Fig. 7. Comparison between stochastic adders using system clock and a local ring-oscillator.

C. Process Size Variation

As the fabrication size decreases, the variation due to fabrication process increases. It is interesting to see if different technology sizes produce different results. To do that evaluation the first test was repeated on an Cyclone IV FPGA which as a different process size, using the same values, where the STRO ran at the same clock frequency as the external clock source. Figure 9 shows that both tests produce similar results. These results suggest that using larger process sizes, the stochastic circuit doesn't benefit much from variability in it's registers, hence behaving similar to the deterministic clock source.

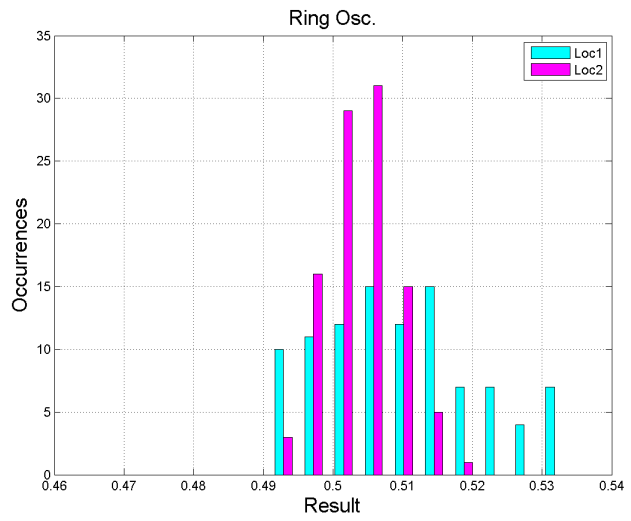

Fig. 8. Comparison between two stochastic adders using the stochastic bitstreams on different locations on the device.

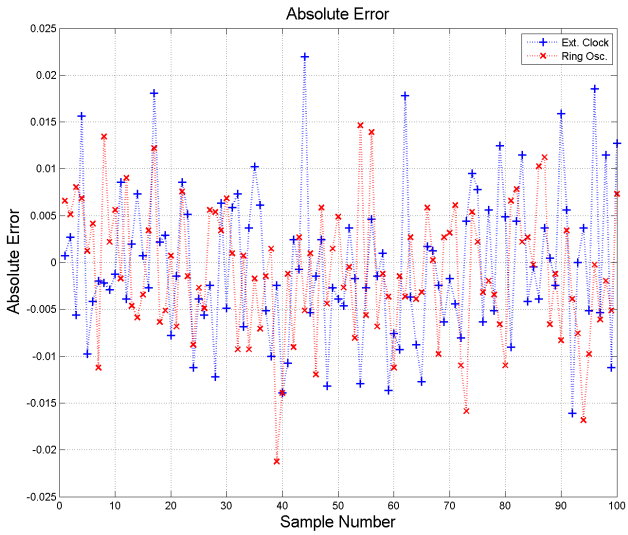

Fig. 9. Comparison between stochastic adders using system clock and a local ring-oscillator on a Cyclone IV FPGA.

D. Placement and Routing Variation

During this research work it was observed that STROs were sensitive to variations in their placement and routing. In some cases these variations represented changes in clock frequency from 18 MHz to 56 MHz, for the same Register Transfer Level (RTL) specification. A possibility to mitigate this problem is a low-level approach where the placement and routing for them is specified. The alternative is a high-level approach which uses LogicLock (LL) regions to contain the placement and limit the variations in routing. It was observed that while changing the size of the LL region, the clock frequencies would also change. Table III illustrates the above with the results for a STRO on a Cyclone IV FPGA.

LL Size (w,h)	1,7	2,4	3,4	4,3	5,2	4,2	7,1
Clk [MHz]	17.2	17.6	18.3	55.9	53.3	53.3	90.0

TABLE III. CLOCK FREQUENCY OF A STRO FOR DIFFERENT LOGICLOCK REGIONS.

VI. CONCLUSIONS AND FUTURE WORK

This work introduces and demonstrates a novel concept of using STRO to drive the clocked signals in stochastic arithmetic circuits, by using process variation to improve their accuracy. This work also makes a contribution on a lightweight test platform for stochastic circuits, including support for automation and the ability to scale to any stochastic circuit size. Results demonstrate benefit in improving the stochastic arithmetic units on devices affected by increased process variation. Future work involves study on the variation of other parameters such as voltage scaling for power optimisation, and on other more complex stochastic systems.

REFERENCES

[1] Z. Guan, J. Wong, S. Chaudhuri, G. Constantinides, and P. Cheung, "A two-stage variation-aware placement method for fpgas exploiting variation maps classification," in *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on*, Aug 2012, pp. 519–522.

[2] B. Gaines, "Stochastic computing systems," A. in Information Systems Science, Ed., vol. 2, 1965, p. 37.

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

[4] M. Suri, O. Bichler, D. Querlioz, G. Palma, E. Vianello, D. Vuillaume, C. Gamrat, and B. DeSalvo, "Cbram devices as binary synapses for low-power stochastic neuromorphic systems: Auditory (cochlea) and visual (retina) cognitive processing applications," in *Electron Devices Meeting (IEDM), 2012 IEEE International*, Dec 2012, pp. 10.3.1–10.3.4.

[5] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm," in *Custom Integrated Circuits Conference (CICC), 2011 IEEE*, Sept 2011, pp. 1–4.

[6] J. von Neumann, "Probabilistic logics and synthesis of reliable organisms from unreliable components," in *Automata Studies*, C. Shannon and J. McCarthy, Eds. Princeton University Press, 1956, pp. 43–98.

[7] Y.-N. Chang and K. Parhi, "Architectures for digital filters using stochastic computing," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, May 2013, pp. 2697–2701.

[8] N. Saraf, K. Bazargan, D. Lilja, and M. Riedel, "IIR filters using stochastic arithmetic," in *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, March 2014, pp. 1–6.

[9] H. Li, D. Zhang, and S. Foo, "A stochastic digital implementation of a neural network controller for small wind turbine systems," *Power Electronics, IEEE Transactions on*, vol. 21, no. 5, pp. 1502–1507, Sept 2006.

[10] S. Tehrani, W. Gross, and S. Mannor, "Stochastic decoding of LDPC codes," *Communications Letters, IEEE*, vol. 10, no. 10, pp. 716–718, Oct 2006.

[11] C. Winstead and S. Howard, "A probabilistic LDPC-coded fault compensation technique for reliable nanoscale computing," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 56, no. 6, pp. 484–488, June 2009.

[12] M. Lin, I. Lebedev, and J. Wawrzynek, "High-throughput bayesian computing machine with reconfigurable hardware," in *Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, ser. FPGA '10. New York, NY, USA: ACM, 2010, pp. 73–82.

[13] F. Zhou, J. Liu, Y. Yu, X. Tian, H. Liu, Y. Hao, S. Zhang, W. Chen, J. Dai, and X. Zheng, "Field-programmable gate array implementation of a probabilistic neural network for motor cortical decoding in rats," *Journal of Neuroscience Methods*, vol. 185, no. 2, pp. 299 – 306, 2010.

[14] B. R. Gaines, "Techniques of identification with the stochastic computer," in *in "Proc. International Federation of Automatic Control Symposium on Identification, Progue*, 1967.

[15] B. Brown and H. Card, "Stochastic neural computation. i. computational elements," *Computers, IEEE Transactions on*, vol. 50, no. 9, pp. 891–905, Sep 2001.

[16] P. Li, D. Lilja, W. Qian, K. Bazargan, and M. Riedel, "The synthesis of complex arithmetic computation on stochastic bit streams using sequential logic," in *Computer-Aided Design (ICCAD), 2012 IEEE/ACM International Conference on*, Nov 2012, pp. 480–487.

[17] J. B. Tenenbaum, E. M. Jonas, and V. K. Mansinghka, "Stochastic digital circuits for probabilistic inference," Massachusetts Institute of Technology, Tech. Rep., November 2008.

[18] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," *ACM Trans. Embed. Comput. Syst.*, vol. 12, no. 2s, pp. 92:1–92:19, May 2013.

[19] W. Qian, M. D. Riedel, H. Zhou, and J. Bruck, "Transforming probabilities with combinational logic," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 30, no. 9, pp. 1279–1292, 2011.

[20] N. Saraf and K. Bazargan, "Sequential logic to transform probabilities," in *Proceedings of the International Conference on Computer-Aided Design*, ser. ICCAD '13. Piscataway, NJ, USA: IEEE Press, 2013, pp. 732–738.

[21] H. Yu, Q. Xu, and P.-W. Leong, "Fine-grained characterization of process variation in FPGAs," in *Field-Programmable Technology (FPT), 2010 International Conference on*, Dec 2010, pp. 138–145.

[22] J. S. Wong and P. Y. Cheung, "Improved delay measurement method in FPGA based on transition probability," in *Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays*, ser. FPGA '11. New York, NY, USA: ACM, 2011, pp. 163–172.

[23] E. Stott, Z. Guan, J. Levine, J. Wong, and P. Cheung, "Variation and reliability in FPGAs," *Design Test, IEEE*, vol. 30, no. 6, pp. 50–59, Dec 2013.

[24] P. Sedcole, J. S. Wong, and P. Y. K. Cheung, "Characterisation of FPGA clock variability," in *Proc. IEEE Computer Society Annual Symp. VLSI ISVLSI '08*, 2008, pp. 322–328.

[25] R. Silwal and M. Niama, "Asynchronous physical unclonable function using FPGA-based self-timed ring oscillator," in *Proceedings of the 22th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, ser. FPGA '14. New York, NY, USA: ACM, 2014, p. 252.

[26] A. Martin and M. Nystrom, "Asynchronous techniques for system-on-chip design," *Proceedings of the IEEE*, vol. 94, no. 6, pp. 1089–1120, June 2006, teste 1.

[27] P. Alfke, "Efficient shift registers, lfsr counters, and long pseudo-random sequence generators," July 1996.

[28] Altera. Cyclone IV device handbook. Online. Altera. [Online]. Available: http://www.altera.com/literature/hb/cyclone-iv/cyclone4-handbook.pdf

[29] ——. Cyclone V device handbook. Online. Altera. [Online]. Available: http://www.altera.com/literature/hb/cyclone-v/cyclone5_handbook.pdf