Lattice FIR Digital Filter Architectures Using Stochastic Computing

Yin Liu and Keshab K. Parhi
University of Minnesota
Department of Electrical and Computer Engineering
Minneapolis, MN, USA

***Abstract*—This paper presents novel architectures for *linear-phase* FIR digital filters using stochastic computing. Stochastic computing systems require fewer logic gates and are inherently fault-tolerant. Thus, these structures are well suited for nanoscale CMOS technologies. Compared to direct-form linear-phase FIR filters, linear-phase lattice filters require twice the number of multipliers but the same number of adders. The hardware complexities of stochastic implementations of linear-phase FIR filters for direct-form and lattice structures are comparable. Using speech signals from ICA '99 Synthetic Benchmarks, it is shown that, for linear-phase FIR filters, the error-to-signal power ratios of stochastic direct-form and stochastic lattice filters are about the same. However, the error-to-signal power of stochastic direct-form or lattice filter is an order of magnitude higher at very low fault rates but is more than two orders of magnitude less when the fault rate is about one percent than the direct-form, where the faults represent random bit-flips at outputs of all logic gates.**

Keywords*—Stochastic computing, stochastic logic, FIR digital filter, lattice structure, linear-phase FIR filters, fault-tolerance, hardware complexity.

I. Introduction

Stochastic computing (SC), first proposed in 1960s [1], has recently regained significant attention due to its fault-tolerance and extremely low-cost of arithmetic units [2] [3]. Despite these advantages, stochastic circuits suffer from long processing latency and degradation of accuracy. Therefore, in the past, SC applications were limited to the fields of neural networks [4] and control [5].

Stochastic numbers are based on unary representation where each bit in the number has the same weight. For example, in a number with 256 bits, if 35 bits are 1, then the value of the number is 35/256. If any bit is flipped, the error introduced due to flipping is 1/256, independent of the location of the flipped bit. In a two's complement number, the error introduced due to flipping of one bit can vary from 1/2 to 1/256, depending on the position of the bit. Thus, the stochastic implementations are inherently fault-tolerant and are less affected by errors due to bit flippings. However, note that an 8-bit binary number requires 256 bits in a stochastic implementation for same resolution.

Recently, we have demonstrated that stochastic computing can be used to design digital filters. We have introduced a new *scaling method* to implement inner products and direct-form FIR digital filters with significantly less output error-to-signal ratio (see [6]). We have also shown that stochastic IIR digital filters could be implemented in lattice structure where the states in filters are *orthogonal* [7].

It is well known that FIR digital filters can be implemented using lattice structures. FIR lattice [8] structures play a central role in the theory of autoregressive signal modeling [9] and are well suited for implementation of adaptive filters. Although, in general, $2N$ multipliers and adders are required for implementation of N-tap FIR lattice filter, *linear-phase* FIR lattice filters require about N multipliers and N adders. Therefore, linear-phase lattice filters can be implemented using approximately the same computation complexity as direct-form linear-phase structures using stochastic computing [10].

The main contributions of this paper are as follows.

- We propose stochastic lattice implementation for *linear-phase* FIR filters. It is shown that it can achieve almost equivalent performance as stochastic implementation of direct-form structures.
- Fault tolerance properties of stochastic lattice FIR digital filters due to random bit-flips at all internal nodes are demonstrated for both direct-form and lattice implementations using speech signals from ICA '99 Synthetic Benchmarks as input [11].

This paper is organized as follows. Section II addresses implementation of stochastic inner-products with input vector size greater than 2. The stochastic lattice implementation of linear-phase FIR filters is proposed in Section III. Section IV presents simulation results, comparison of hardware resource and fault-tolerance analysis of lattice and direct-form implementations of linear-phase FIR filters using stochastic logic.

II. Stochastic Digital filters

A. SC Inner-Product

Inner-products are fundamental components for stochastic digital filter design. They break the range constraint of coefficients ([-1,1]) by integrating the ratio of a and b into selecting signal of the multiplexer, which makes it possible to implement stochastic digital filters with any arbitrary coefficients (see [6]). In [6], only the architecture of SC inner-product with input vector of size 2 was introduced. Fig. 1 presents the architecture of an SC inner-product whose input vector size is greater than 2. Consider the computation of the inner-

This research was supported by the National Science Foundation under grant number CCF-1319107.

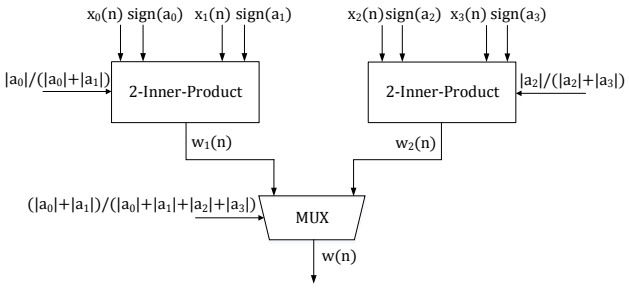

Fig. 1: The architecture of a stochastic inner-product with input vector size of 4.

product $< (a_0, a_1, a_2, a_3) \cdot (x_0(n), x_1(n), x_2(n), x_3(n)) >$. The internal nodes are described by:

$$\begin{cases} w_1(n) = \frac{1}{|a_0|+|a_1|}(a_0x_0(n) + a_1x_1(n)) \\ w_2(n) = \frac{1}{|a_2|+|a_3|}(a_2x_2(n) + a_3x_3(n)) \end{cases}$$

The final output is

$$w(n) = \frac{a_0x_0(n) + a_1x_1(n) + a_2x_2(n) + a_3x_3(n)}{|a_0| + |a_1| + |a_2| + |a_3|}$$

Notice that the output result is scaled by $\frac{1}{|a_0|+|a_1|+|a_2|+|a_3|}$.

In the second level of tree structure, we need to compute $((|a_0| + |a_1|), (|a_2| + |a_3|)) \cdot (w_1(n), w_2(n))$. Since the coefficients $(|a_0| + |a_1|)$ and $(|a_2| + |a_3|)$ are positive, the XNOR gates in the 2-input inner-product are not necessary. Therefore, only nodes at the first level of the tree require full implementation of 2-input inner-products. Other nodes can be implemented using single multiplexers.

B. Implementation Considerations for Stochastic Digital Filters

1) Trade-off in delay element implementations: In [6], stochastic FIR filters in direct-form were implemented using SC inner-product module based on two approaches. Fig. 2(a) shows one approach where the input signal $x(n)$ is first converted into a stochastic bit-stream, and then is passed through the delay line. In Fig. 2(b), the input signal first passes through the delay line, and then each signal from the delay line is converted separately to a stochastic bit sequence.

TABLE I: Area consumption comparison of two implementations for stochastic direct-form FIR filters in terms of equivalent 2-input NAND gates.

Type of Implementations	Filter Order		
	2	4	6
2's complement	3243	6575	9147
Type-I	25761	51107	76450
Type-II	1453	2445	3762

Table I shows synthesis results of two implementations for stochastic direct-form FIR filters. We assume that binary word-length is 10, whereas the length of stochastic sequence is 1024. The consumptions of area are given in terms of equivalent two input NAND gates in 65nm libraries. Type-I corresponds to the architecture in Fig. 2(a) and Type-II represents the

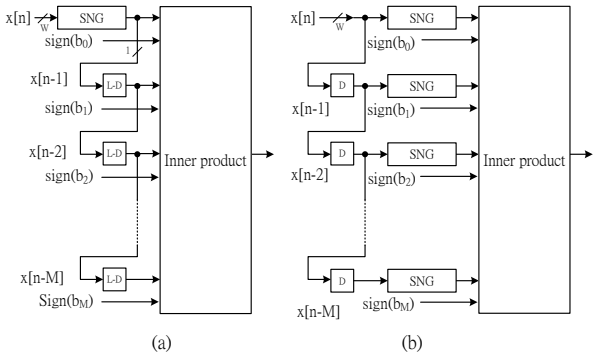

Fig. 2: Two approaches to delaying input signals in stochastic digital filters: the input samples are delayed in (a) stochastic representation, (b) binary representation.

architecture shown in Fig. 2(b). Type-I architecture leads to 10-fold increase in hardware complexity, compared to Type-II architecture, and it is even greater than traditional 2's complement filters. This fact suggests that in a feasible architecture of any kind of stochastic digital filters, signals should be stored in delay elements in 2's complement format even though more SNGs are required.

2) Hardware efficiency of stochastic digital signal processing system: In stochastic DSP implementations, the complexity of an addition, that is, the cost of a multiplexer containing SNGs, is significantly higher than that of an XNOR gate which implements a multiplication. Therefore, the optimization of stochastic filter architectures should focus on reducing the number of additions in system.

III. Stochastic lattice implementation of linear-phase FIR filters

A typical lattice FIR filter is shown in Fig. 3. Notice that $2N$ adders are required for N-tap FIR lattice filter while a direct-form FIR filter with the same order has only N adders. It means the number of inner-products in stochastic lattice implementations is twice as that of stochastic direct-form FIR filters. Thus, hardware complexity and noise will increase due to the increase in the number of computations.

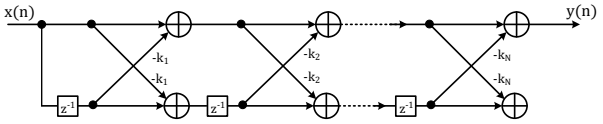

Fig. 3: The block diagram of an N-tap FIR lattice filter.

However, lattice structure of linear phase FIR filters can be implemented with same number of computations as direct-form structure. Assume that k_i's represent coefficients in lattice structure. Directly applying coefficients-to-k-parameter algorithm [12] for linear-phase FIR filters leads to singularity which is caused by the symmetry of linear-phase FIR coefficients [10]. Fig. 4 shows an alternative approach to implementing lattice structure for an N-tap linear-phase FIR filter, where $L = \lfloor \frac{N+1}{2} \rfloor$ and $M = \lfloor \frac{N}{2} \rfloor$.

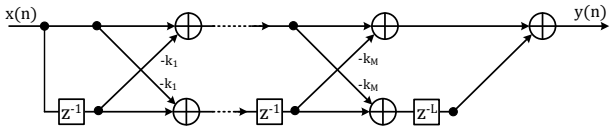

Fig. 4: The block diagram of an N-tap linear-phase FIR lattice filter.

Assume the linear-phase FIR filter is described as $y(n) = b_0x(n) + b_1x(n-1) + \cdots + b_Nx(n-N)$, where $b_i = b_{N-i}$. The key idea is applying Schur algorithm [13] only for $[b_0, b_1, \cdots, b_{\lfloor \frac{N+1}{2} \rfloor}]$ rather than all $N+1$ coefficients to avoid the singularity where $k_i = \pm 1$ (see [10] for detailed derivation).

An N-tap stochastic lattice implementation for linear-phase FIR filter is described in Fig. 5. It computes the scaled result $y(n)/(2\prod_{i=1}^{m}(1+|k_i|))$. Coefficients $s(k_i)$ represent the sign of k_i. Full implementations of SC inner-products are not required since out of four coefficients in a lattice stage two are always unity. Stochastic-to-binary(S2B) modules [6] are used to convert stochastic bit-streams to binary numbers. The size of each delay element is determined by the word-length of 2's complement representation. All coefficients in the architecture are represented by stochastic sequences. Unlike stochastic lattice implementation of IIR filters [7], coefficients do not require extra scaling since computation results of top line and bottom line in a lattice stage are equivalently scaled by SC inner-product modules.

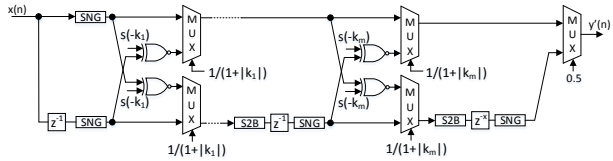

Fig. 5: The architecture of a stochastic implementation for an N-tap linear-phase FIR lattice filter.

IV. Experimental Results

In this section, we present the experimental results for stochastic direct-form implementation and lattice implementation for linear-phase FIR filters. The metrics of measurement include fault-tolerance performance, and hardware complexity.

A. Simulation Results

A truncated speech signal from ICA '99 Synthetic Benchmarks is used as the input signal. In our simulation, the length of the stochastic sequence is 1024 and the corresponding word-length of 2's complement representation is 10. A total of 5000 input samples are used for simulation.

Fig. 6 shows the spectrums of input and output signals obtained from stochastic and ideal filters for a 7^{th}-order linear-phase FIR filter with cut-off frequency at 0.1π. Table II shows the error-to-signal power ratio for stochastic direct-form and lattice implementation of low-pass linear-phase FIR filters with different orders and cut-off frequencies. From simulation results, we observe that they have equivalent performance.

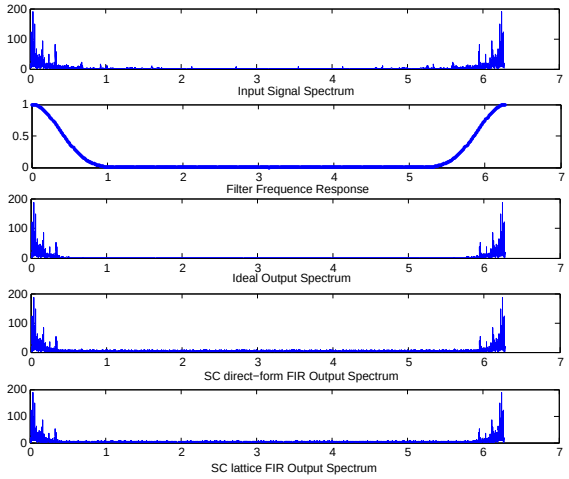

Fig. 6: The spectrums of input signal, filter frequency response, ideal output, stochastic direct-form output, and stochastic lattice output for filtering using a 7^{th}-order linear-phase FIR filters with cut-off frequency at 0.1π.

TABLE II: The output error to signal power ratio for stochastic linear-phase FIR filters of different orders and cut-off frequencies.

(a) Stochastic direct-form implementation

Filter Order	Low-pass Cut-Off Frequency			
	0.1π	0.3π	0.5π	0.7π
3	0.0263	0.0250	0.0230	0.0221
5	0.0345	0.0313	0.0294	0.0266
7	0.0387	0.0346	0.0367	0.0362

(b) Stochastic lattice implementation

Filter Order	Low-pass Cut-Off Frequency			
	0.1π	0.3π	0.5π	0.7π
3	0.0261	0.0249	0.0240	0.0225
5	0.0338	0.0319	0.0286	0.0289
7	0.0401	0.0350	0.0416	0.0437

B. Hardware Complexity

The area consumption of stochastic FIR filters are evaluated using 65nm technology. The architectures are synthesized using Synopsys Design Compiler. We also compare hardware complexity between traditional binary implementations and stochastic implementations. In stochastic implementations, the length of stochastic sequences is 1024, and binary numbers in traditional implementations require 10 bits. Table III shows hardware complexity of binary and stochastic implementations for FIR filters.

TABLE III: The hardware complexity comparison of implementations for FIR filters in terms of equivalent 2-NAND gates.

Type of Implementations	Filter Order		
	3	5	7
2's complement	4573	7941	10593
stochastic direct-form (Type-II)	2091	3193	4186
stochastic lattice	1848	2716	3566

The results show that stochastic implementations require less hardware resources than traditional binary implementation due to the low cost of arithmetic units. Moreover, stochastic lattice implementations for linear-phase FIR filters consume less hardware resources than stochastic direct-form implementations. Comparing Fig. 2(b) and Fig. 5, we can observe that for an N-tap linear-phase FIR filter, a stochastic direct-form implementation requires N 2-input inner-products while a stochastic lattice implementation requires $(2 \cdot \lfloor \frac{N}{2} \rfloor + 1)$ 2-input inner-product. The hardware complexities of the additions in two implementations are about the same. However, there are N SNG modules in Type-II stochastic direct-form implementation, whereas stochastic lattice implementation requires $(\lfloor \frac{N}{2} \rfloor + 1)$ SNG modules. Compared to SNG modules, hardware complexity of a S2B module can be ignored. Therefore, the low hardware complexity of stochastic lattice implementation is explained by the reduction of the number of SNG modules.

C. Fault Tolerance Analysis

We performed fault-tolerance test for both stochastic FIR filters by randomly injecting bit-flipping error at all internal nodes and measuring the corresponding output error-to-signal power ratio for each implementation. Real speech signals from ICA '99 Synthetic Benchmarks are used as the test inputs. The length of the stochastic sequence is 1024. A total of 5000 input samples are used. We control the level of injected soft error by flipping certain percent bits of all internal computational nodes in circuits. Flipped bits are selected at random.

A 7-tap linear-phase FIR filter with cut-off frequency at 0.1π is considered. The signals at marked nodes in Fig. 7 are flipped for a specified percent at random. A total of 14 internal nodes are considered in traditional binary and stochastic direct-form implementations. The stochastic lattice implementation has 12 internal computational nodes. Table IV and Fig. 8 present output error-to-signal power ratios due to bit-flipping.

TABLE IV: The output error to signal power ratios due to random bit-flipping for different implementations for a 7^{th}-order low-pass linear-phase FIR filter with cut-off frequency 0.1π.

Type of Implementations	Percentage of Bit-flipping					
	0%	0.01%	0.05%	0.1%	0.5%	1%
traditional binary	0.0038	0.1043	0.5445	1.0917	5.0803	9.0240
stochastic direct-form	0.0465	0.0464	0.0462	0.0467	0.0488	0.0540
stochastic lattice	0.0470	0.0472	0.0476	0.0478	0.0563	0.0820

It is shown that bit-flipping almost has no impact on the output accuracy of stochastic direct-form and lattice implementations when flipping percentage is under 0.5%. Starting with 0.01% bit-flipping, the performance of the traditional binary implementation is degraded significantly due to random bit-flippings. The traditional implementation has an order of magnitude less output error-to-signal power for very low rate of bit-flippings but its output error-to-output ratio is more than two orders of magnitude higher at fault rates of about 1%.

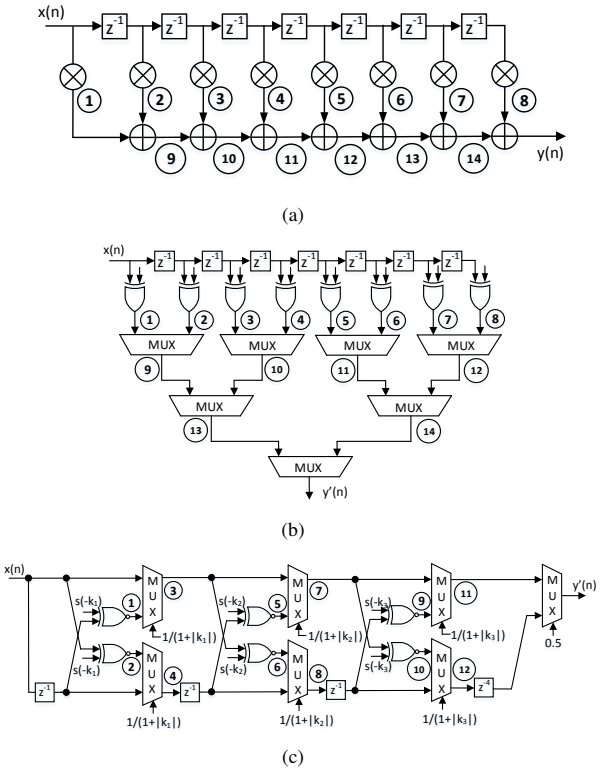

Fig. 7: The architectures for (a) traditional binary FIR filter, (b) stochastic direct-form FIR filter and (c) stochastic lattice implementation of linear-phase FIR filter, where random bit-flippings occur at the nodes marked (SNG and S2B modules are not shown in this figure).

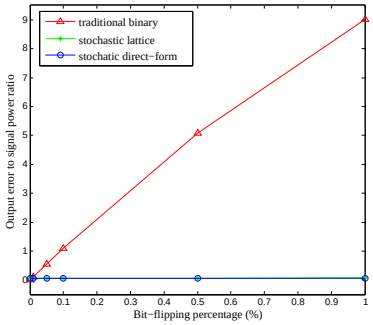

Fig. 8: Fault-tolerance test results of different implementations for a 7^{th}-order low-pass linear-phase FIR filters with cut-off frequency 0.1π.

V. CONCLUSION

This paper has presented a novel implementation of a linear-phase FIR digital filter using stochastic logic. Future work will investigate a theoretical analysis of error-to-signal ratio of the stochastic implementation. Future work will also be directed towards comparison of the fault tolerance of the proposed architecture with fault-tolerant two's complement architectures using partial triple modular redundancy [14].

REFERENCES

[1] B. R. Gaines, "Stochastic computing," in *Proceedings of AFIPS spring joint computer conference*, pp. 149–156, ACM, 1967.

[2] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," *ACM Transactions on Embedded computing systems (TECS)*, vol. 12, no. 2s, p. 92, 2013.

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

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

[5] A. Dinu, M. Cirstea, and M. McCormick, "Stochastic implementation of motor controllers," in *Proceedings of the 2002 IEEE International Symposium on Industrial Electronics (ISIE)*, vol. 2, pp. 639–644, 2002.

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

[7] K. K. Parhi and Y. Liu, "Architectures for IIR digital filters using stochastic computing," in *Proceedings of 2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 373–376, 2014.

[8] A. Gray Jr. and J. Markel, "Digital lattice and ladder filter synthesis," *IEEE Transactions on Audio and Electroacoustics*, vol. 21, no. 6, pp. 491–500, 1973.

[9] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*. Wiley 1999.

[10] K. Schwarz, "Linear phase FIR-filter in lattice structure," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), 1993*, pp. 347–350, 1993.

[11] "ICA'99 synthetic benchmarks." http://sound.media.mit.edu/ica-bench/, Sept. 2014.

[12] A. V. Oppenheim, R. W. Schafer, J. R. Buck, *et al.*, *Discrete-time signal processing*, vol. 2. Prentice-hall Englewood Cliffs, 1989.

[13] J. Schur, "Über potenzreihen, die im innern des einheitskreises beschränkt sind.," *Journal für die reine und angewandte Mathematik*, vol. 147, pp. 205–232, 1917.

[14] R. Parhi, C. H. Kim, and K. K. Parhi, "Fault-tolerant ripple-carry binary adder using partial triple modular redundancy (PTMR)," in *Proceedings of 2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015.