Design, Evaluation and Fault-Tolerance Analysis of Stochastic FIR Filters

Ran Wang, Jie Han, Bruce F. Cockburn, and Duncan G.. Elliott
Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada

***Abstract*— Stochastic computing utilizes compact arithmetic circuits that can potentially lower the implementation cost in silicon area. In addition, stochastic computing provides inherent fault tolerance at the cost of a less efficient signal encoding. Finite impulse response (FIR) filters are key elements in digital signal processing (DSP) due to their linear phase-frequency response. In this article, we consider the problem of implementing FIR filters using the stochastic approach. Novel stochastic FIR filter designs based on multiplexers are proposed and compared to conventional binary designs implemented using Synopsys tools with a 28-nm cell library. Silicon area, power and maximum clock frequency are obtained to evaluate the throughput per area (TPA) and the energy per operation (EPO). For equivalent filtering performance, the stochastic FIR filters underperform in terms of TPA and EPO compared to the conventional binary design, although the stochastic design shows more graceful degradation in performance with a significant reduction in energy consumption. A detailed analysis is performed to evaluate the accuracy of stochastic FIR filters and to determine the required stochastic sequence length. The fault-tolerance of the stochastic design is compared with that of the binary circuit enhanced with triple modular redundancy (TMR). The stochastic designs are more reliable than the conventional binary design and its TMR implementation with unreliable voters, but they are less reliable than the binary TMR implementation when the voters are fault-free.**

Keywords—stochastic computing, FIR filter, throughput per area, energy per operation, stochastic sequence length, fault tolerance

NOTATION

$P(S)$: probability encoded by a stochastic sequence S
N_b: the bit resolution for the binary filter
N_s: the number of bits in stochastic sequence S
N_f: the number of taps in the FIR filter
X: a binary number, signed or unsigned
$X[n]$: the binary value of a time-series variable X at time n
$S(X)$: a stochastic binary sequence encoding the probability $\frac{X}{L}$, where L is the sequence length.

I. Introduction

THE importance of finite impulse response (FIR) filters in digital signal processing and the potential benefits of stochastic computing motivated us to investigate the possibility of implementing stochastic FIR filters. Both manufacturing variations and transient errors pose additional challenges to reliable operation. Stochastic computing methods [1, 2] can be exploited to address the above issues and thus possibly allow operation with less reliable, leading edge processes in very low voltage and/or high noise operating conditions.

In image processing, Li and Lilja showed that stochastic circuits can outperform conventional binary designs for key image processing algorithms with respect to important design metrics [3]. Specifically, sequential stochastic computational elements were built using finite state machines. Interesting results for a stochastic implementation of the kernel density estimation (KDE)-based image segmentation algorithm are reported in [4]. Alaghi *et al.* investigated an edge-detection algorithm for real-time image processing [5]. It was shown that the area-delay product of the stochastic edge detection circuit is only 8.7% of that of a conventional binary circuit. Qian and Riedel compared stochastic hardware implementations of polynomial arithmetic [6]. Chang and Parhi investigated novel designs for both FIR and infinite impulse response (IIR) filters based on stochastic logic [7]. Several low-pass and high-pass filters with different cut-off frequencies were considered. In those filters, the coefficients are encoded as stochastic selection signals of multiplexers and XOR gates are used to invert the inputs when the corresponding coefficients are negative numbers. Our new approach works with both unsigned and signed stochastic inputs directly without additional operations, while requiring additional multiplexers as weight generators. Another stochastic FIR filter was designed using a wire selecting method in [8]. The authors claimed that the proposed stochastic FIR filter is more cost-efficient than traditional filters in terms of hardware resource for less than 9-bit implementations. However, it is not clear whether or not the stochastic filter can effectively match the performance of the corresponding traditional filters. In [9], the authors proposed to modify conventional ADCs to generate analog to stochastic streams with minimal overhead to implement IIR filters. Input to output frequency response spectra were computed to demonstrate the idea. However, this method is limited to IIR filters where feedback networks are needed.

In this paper, three different stochastic FIR filter designs are investigated. The conventional weighted average (CWA) design exploits basic stochastic arithmetic elements, such as the XNOR gate for multiplication and the multiplexer for addition. In the hardwired weighted average (HWA) design, the filter coefficients or weights are given by repeating inputs to the multiplexer. Finally, the multi-level weighted average (MWA) design leverages the fact that every input signal is selected with a certain weight determined by the selection inputs to a multiplexer. The MWA design explicitly uses additional multiplexers to generate filter coefficients as the weights to a weighted adder. In all three designs, the weight of an input signal is the probability of selecting that input signal. That

probability is then encoded in the frequency at which the corresponding combination of the selecting signals occurs in the bit streams. It is shown that both HWA- and MWA-based FIR filters have improved performance in terms of area, power and speed, compared to the CWA design.

Different resolutions are considered to determine the threshold (or break-even point) that defines the competitive resolution range for stochastic circuits. 3-bit to 16-bit FIR filters using both the stochastic and binary approaches are implemented initially. Then the minimum required stochastic sequence length that enables the stochastic circuit to filter signals as accurately as the binary circuit is determined empirically. The general metrics of throughput per area (TPA) and energy per operation (EPO) are used to characterize and compare the performance of the stochastic and conventional circuits.

This article is a significant extension of [10]. It contains the following main contributions.

- Stochastic FIR filters based on the HWA and MWA designs are proposed by using multiplexers to implement weighted adders. Different strategies are considered for generating the filter coefficients.
- A detailed analysis is performed to evaluate the accuracy of the binary and stochastic filters. The analytical results provide an estimation of the minimum stochastic sequence length that is required to ensure that the performance of the stochastic filter matches that of a conventional filter.
- A detailed comparison is provided with respect to the fault tolerance of the HWA- and MWA-based stochastic filters. The conventional binary filter and its fault-tolerant triple modular redundancy (TMR) implementation are considered in the comparison.

In the sequel, we first review stochastic computing and binary FIR filter designs in Section II. In Section III, the stochastic FIR filter designs are presented. In Section IV, the performance is compared by considering the magnitude responses of both filters. In Section V, the simulation results from a Synopsys synthesis tool are reported. In Section VI, the accuracy of the FIR filters using both binary and stochastic computing is assessed. The fault tolerance of stochastic and conventional binary circuits are determined and compared in Section VII. Section VIII concludes the paper.

II. Background

A. Stochastic Computing

Stochastic computing involves processing numbers that are encoded as real values, which are represented using stochastic bit-streams. If there are N_1 1's in a bit stream containing N_s bits, where $N_1 \leq N_s$, it represents either the (unsigned) unipolar number N_1/N_s or the (signed) bipolar number $(2N_1–N_s)/N_s$ [1, 2]. For example, "0001100101" denotes 2/5 in unipolar and -1/5 in bipolar for $N_s = 10$ and $N_1 = 4$. Stochastic computing elements can often be built using very small circuits with low power consumption [1]. To encode a binary number containing N_b bits, the minimum sequence length is $N_{s,min} = 2^{N_b}$. However, the required sequence length N_s is usually made larger for increased accuracy during stochastic processing. Therefore, a performance matching multiplier is introduced as $PMM = N_s/2^{N_b}$. The stochastic sequences are usually generated using linear feedback shift registers (LFSRs) or other similar pseudo-random bit sequence generators. Fig. 1 shows a general stochastic computing system [2].

A stochastic weighted adder can be implemented with a multiplexer. If data inputs X_1 and X_2 are both encoded as unipolar (or bipolar) stochastic sequences, then the output Y of a two-input multiplexer will also be unipolar (bipolar). However, the multiplexer selecting signal A must be encoded as a unipolar sequence. Consider a stochastic implementation of the weighted sum

$$Y = A_1X_1 + A_2X_2, \quad (1)$$

where A_1 and A_2 are positive weights whose sum is one (i.e., $A_1 + A_2 = 1$ and $A_1, A_2 > 0$). Data inputs X_1 and X_2 are encoded as stochastic sequences $S(X_1)$ and $S(X_2)$, and weights A_1 and A_2 are encoded as unipolar stochastic sequences $S(A_1)$ and $S(A_2)$. For the multiplexer, the select input A is driven by unipolar sequence $S(A_1)$, and so this input is 1 with probability $P(S(A_1))$, i.e.,

$$P(S(A_1)) = A_1 = 1 - A_2. \quad (2)$$

The multiplexer output Y will be 1 with probability $P(S(Y))$, i.e.,

$$P(S(Y)) = A_1 \cdot P(S(X_1)) + A_2 \cdot P(S(X_2)). \quad (3)$$

First consider the case of unipolar data inputs. By definition the numbers encoded by sequences $S(X_1)$, $S(X_2)$ and $S(Y)$ are exactly $P(S(X_1)) = X_1$, $P(S(X_2)) = X_2$ and $P(S(Y)) = Y$, respectively. Thus the computed output value Y will be the desired weighted sum $A_1X_1 + A_2X_2$.

Now consider the case of bipolar data inputs. By definition, the numbers encoded by $S(X_1)$, $S(X_2)$ and $S(Y)$ are exactly $\frac{X_1+1}{2}$, $\frac{X_2+1}{2}$ and $\frac{Y+1}{2}$, respectively. Thus the output sequence $S(Y)$ will be 1 with probability $P(S(Y))$, i.e.,

$$P(S(Y)) = \frac{A_1X_1+A_2X_2+A_1+A_2}{2} = \frac{A_1X_1+A_2X_2+1}{2}. \quad (4)$$

From the definition of the bipolar encoding, sequence $S(Y)$ encodes the number

$$Y = 2 \cdot P(S(Y)) - 1 = A_1X_1 + A_2X_2, \quad (5)$$

which is again the desired weighted sum.

The same argument can be generalized to weighted sums containing $N \geq 2$ inputs implemented with N-input multiplexers, where the weights $A_1, A_2, \ldots, A_N$ sum up to 1.

Fig. 1. A basic stochastic computing system.

B. Encoding Numbers as Unipolar and Bipolar Stochastic Sequences

Stochastic number generators (SNGs) are typically based on pseudo-random bit generators such as linear feedback shift registers (LFSRs). For example, to generate the stochastic sequence for a 4-bit unsigned binary number, the SNG in Fig. 2(a) is implemented with a 4-bit LFSR [2]. The SNG in Fig. 2(a) converts a 4-bit unsigned binary number x to a stochastic number (sequence) of length 16. The all-zero state must be inserted into the maximum-length (15-state) nonzero state sequence by adding extra combinational logic to a traditional 4-bit LFSR (see Fig. 3) [2]. The SNG takes advantage of weight generation. The bit streams named $W3$, $W2$, $W1$ and $W0$ represent the weights 1/2, 1/4, 1/8 and 1/16, respectively. The

binary number x is converted bit-by-bit with different weights assigned to them. Therefore, we have

$$P(S) = \frac{1}{2}\cdot x[3] + \frac{1}{4}\cdot x[2] + \frac{1}{8}\cdot x[1] + \frac{1}{16}\cdot x[0] = \frac{(8\cdot x[3] + 4\cdot x[2] + 2\cdot x[1] + 1\cdot x[0])}{16} = x/16, \quad (13)$$

where S is the output sequence of the SNG and $P(S)$ is the probability that S represents. Thus S is the stochastic representation of the binary number x.

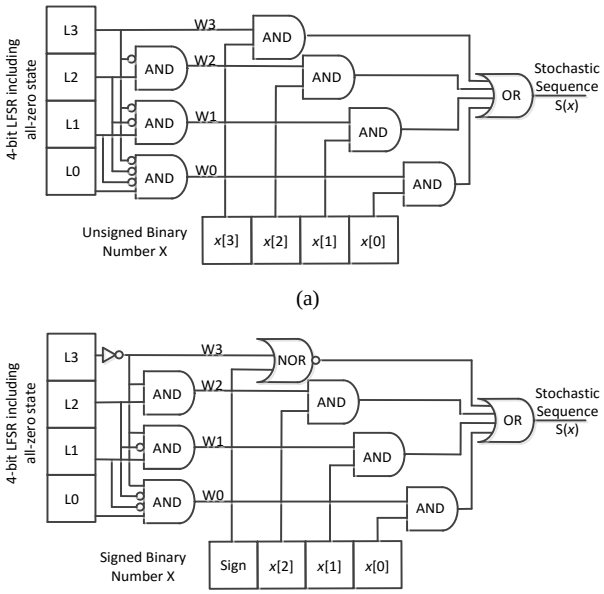


Fig. 2. (a) Unipolar stochastic number generator [2] and (b) Bipolar stochastic number generator.

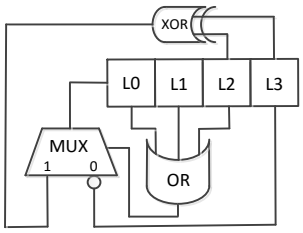


Fig. 3. A 4-bit LFSR with the all-zero state.

For signed numbers, we use bipolar stochastic representations [1]. An N_s-bit stochastic sequence with N_1 1's encodes the probability of $(2\times N_1 - N_s)/N_s$. To design an SNG for signed numbers, let us consider the mappings of a signed binary number to its stochastic representation. For example, for each 4-bit signed binary number in two's complement, Table 1 shows the relationship with the probability that every single bit in the stochastic sequence is '1' and the probability that is encoded in the bipolar stochastic representation, assuming that the sequence length is 16 bits. This relationship reveals that the stochastic conversion of a signed binary number can be implemented by the SNG for unsigned numbers by simply inverting the sign bit and treating the remaining bits in the signed binary number as for an unsigned number. This SNG design is shown in Fig. 2(b). To invert the signal of the sign bit in the 4-bit signed number, a NOR gate is used to replace the AND gate connected to the sign bit. Also, some inverters are combined into one at the output of L3.

Table 1. The mapping scheme for unsigned binary numbers and their corresponding stochastic representations.

Decimal	Signed Binary Number in 2's complement	Probability of any bit being '1' in the 16-bit sequence	Probability in bipolar representation: $(2\times N_1 - N_s)/N_s$
7	0111	15/16	(2×15−16)/16 = 7/8
6	0110	14/16	(2×14−16)/16 = 6/8
5	0101	13/16	(2×13−16)/16 = 5/8
4	0100	12/16	(2×12−16)/16 = 4/8
3	0011	11/16	(2×11−16)/16 = 3/8
2	0010	10/16	(2×10−16)/16 = 2/8
1	0001	9/16	(2×9−16)/16 = 1/8
0	0000	8/16	(2×8−16)/16 = 0/8
−1	1111	7/16	(2×7−16)/16 = −1/8
−2	1110	6/16	(2×6−16)/16 = −2/8
−3	1101	5/16	(2×5−16)/16 = −3/8
−4	1100	4/16	(2×4−16)/16 = −4/8
−5	1011	3/16	(2×3−16)/16 = −5/8
−6	1010	2/16	(2×2−16)/16 = −6/8
−7	1001	1/16	(2×1−16)/16 = −7/8
−8	1000	0/16	(2×0−16)/16 = −8/8

C. FIR Filters

An N_f –tap FIR filter implements a sum of products over a sliding window of the N_f most recent input samples, as specified in (6). The coefficients $H[i]$ (i = 0, 1, …, $N_f - 1$) give the finite impulse response of the filter.

$$Y[n] = \sum_{i=0}^{N_f-1} H[i]\, X[n-i] \quad (6)$$

The hardware implementation of an FIR filter consists of adders and multipliers as well as delay units, which are typically implemented as D flip flops (DFFs) (Fig. 4). To meet the minimum resolution requirement in a stochastic design, the stochastic sequence must be of length $\geq PMM \cdot 2^{N_b}$, where N_b is the binary bit resolution and $PMM \geq 1$. However, this requires huge storage and almost certainly excessive latency [7].

One solution to the relatively long latency and large storage cost is to move the binary input signal samples through the DFFs before they are expanded into stochastic bit streams (Fig. 5). At every stochastic clock cycle there will be one stochastic bit generated by each of the SNGs. These stochastic bits and the stochastic coefficients are then processed serially to form the final output. In Fig. 5, $S(X[n\text{-}i])$ and $S(H[i])$ are the stochastic bit streams encoding the values of $X[n\text{-}i]$ and $H[i]$, respectively, where i = 0, 1, 2, 3. An N_s-bit stochastic sequence $S(Y[n])$ is produced as the filter output over N_s stochastic clock cycles. Note that the sample clock cycle contains N_s stochastic clock cycles to allow one stochastic operation. The design in Fig. 5 requires four expensive SNG modules (based on four N_b-bit LFSRs) but only three N_b-bit registers. Therefore, it has relatively low cost and thus is chosen for further investigation.

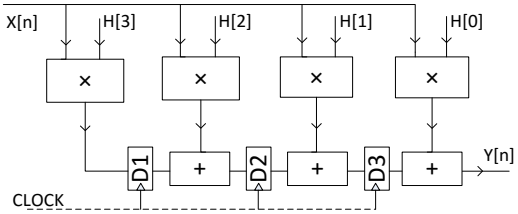


Fig. 4. A 4-tap FIR filter design (i.e., N_f = 4 in (6)).

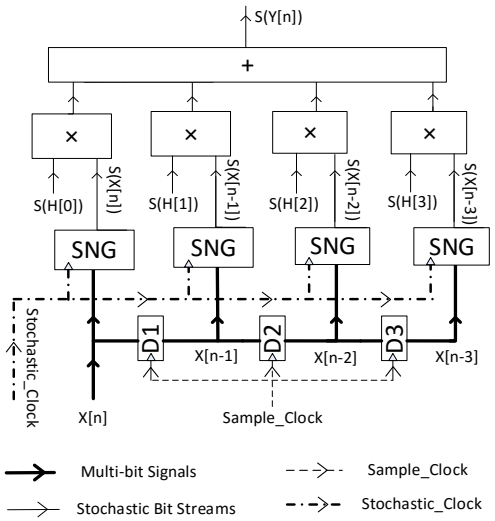


Fig. 5. A 4-tap Stochastic FIR filter design (i.e., N_f = 4 in (6)) [7].

III. Proposed Stochastic FIR Filters

A. Conventional Weighted Average (CWA) Design

The conventional weighted average (CWA) design was built using conventional stochastic arithmetic elements as a basis for comparison with two novel stochastic designs introduced later. At the core of an N_f-tap FIR filter is an N_f-input weighted average function. For a 16-tap FIR filter, this function is implemented using stochastic logic, see Fig. 6. The multiplexer (MUX) is used as a simple adder. The XNOR gates implement bipolar multiplications provided that the two input sequences, i.e., the input sequence and the corresponding coefficient sequence, are statistically independent [2]. Two N_s-bit bipolar stochastic sequences *S1* and *S2* are said to be independent if

$$P(\overline{S1 \oplus S2}) = P(S1) \cdot P(S2) \qquad (7)$$

where *P(S1)* and *P(S2)* denote the probabilities encoded by *S1* and *S2*, respectively. $\overline{S1 \oplus S2}$ denotes the sequence produced by an XNOR gate with *S1* and *S2* as input sequences. Note that the selecting signals are unipolar sequences encoding the probability of 0.5. In Fig. 6, all the numbers to data inputs are converted by using bipolar SNGs (denoted by SNG_b) and all the numbers to selecting inputs are converted using unipolar SNGs (denoted by SNG_u).

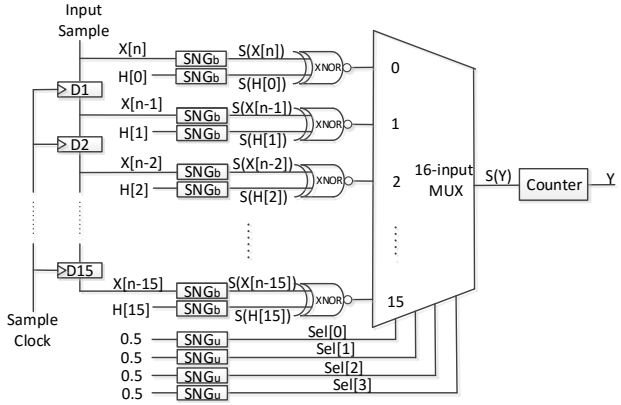


Fig. 6. The 16-tap stochastic FIR filter using the conventional stochastic design.

B. Hard-wired Weighted Average (HWA) Design

In the CWA design, the SNGs cannot be shared, due to the requirement of signal independency. In the hard-wired weighted average (HWA) design, however, the absolute values of the coefficients can be implemented by assigning unbiased stochastic sequences to the selecting inputs of the multiplexer. In an unbiased stochastic sequence, the probabilities of each bit being '1' and '0' are the same, i.e., 0.5. The probability is then the same for selecting each of the inputs. However, a particular data input can be given more weight in the multiplexer output by connecting the input to multiple multiplexer inputs. Note that the signs of the coefficients can be implemented by XOR gates at the data inputs of the multiplexer. XOR gates invert the corresponding input when the coefficient is negative. When the coefficients are positive, the XOR gates become buffers.

In Fig. 7, for example, Wires 8 to 15 are associated with the same input $S(X[\text{n-4}])$, where $S(X[\text{n-4}])$ is the stochastic bit stream encoding the value of $X[\text{n-4}]$. Thus the probability of selecting the input $S(X[\text{n-4}])$ is 8/16 or 1/2, which means the coefficient of $X[\text{n-4}]$ is either 1/2 or -1/2. Similarly, all the other coefficients can be weighted by repeating inputs appropriately. The weighted average function in (8) requires a multiplexer with four selecting inputs. It can be implemented by a 16-input multiplexer with combined data inputs as in Fig. 7.

$$Y = sign(A[0]) \cdot \tfrac{1}{16} \cdot X[\text{n}] + sign(A[1]) \cdot \tfrac{1}{16} \cdot X[\text{n}-1] + sign(A[2]) \cdot \tfrac{1}{8} \cdot X[\text{n}-2] + sign(A[1]) \cdot \tfrac{1}{4} \cdot X[\text{n}-3] + sign(A[0]) \cdot \tfrac{1}{2} \cdot X[\text{n}-4]. \qquad (8)$$

The HWA design potentially improves upon the CWA design in that the SNGs for the weights can be removed.

In general, to implement the N_f-tap FIR filter in (6) using an N_b-bit resolution, the major steps are as follows:

1) Convert the floating point coefficients $H[i]$ in (6) to fixed point N_b-bit binary numbers $A[\text{n}-i]$, where i = 0, 1, …, $N_f - 1$.

2) Calculate the sum of all the absolute values of the coefficients $A = \sum_{i=0}^{N_f-1} |A[\text{n}-i]|$ where $A[\text{n}-i]$ is an N_b-bit binary number, for i = 0, 1, …, $N_f - 1$.

3) The multiplexer has 2^{N_m} data inputs and N_m selecting inputs, where N_m is determined by $N_m = \lceil log_2 A \rceil$ and $\lceil \cdot \rceil$ is the ceiling function. Each selecting input is an unbiased stochastic sequence encoding the probability of 0.5.

4) The number of inputs to be combined is given by $|A[i]|$ for input $X[\text{n}-i]$ (i = 0, 1, ..., $N_f - 1$). The sign of the coefficient A[i] (i = 0, 1, ..., $N_f - 1$) is one of the inputs of the corresponding XOR gate.

5) Use a synthesis tool to optimize the design.

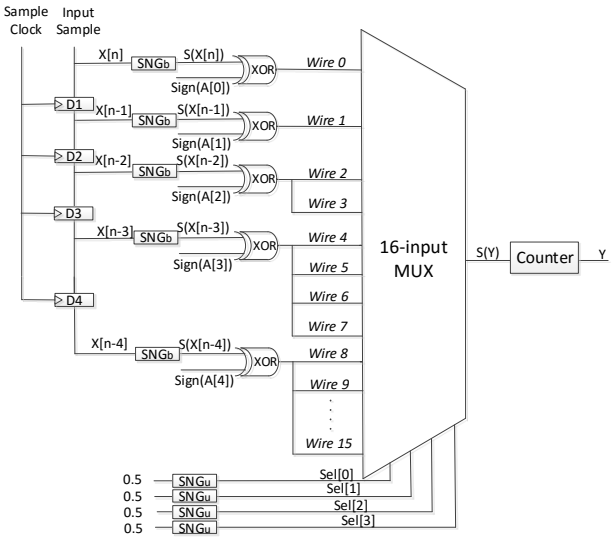


Fig. 7. The hard-wired weighted average design of a 5-tap FIR filter.

C. *Multi-level Weighted Average (MWA) Design*

In general, the function of a weighted adder is given by

$$Y = \alpha \sum_{i=0}^{N_f-1} A[i]\, X[n-i], \tag{9}$$

where $\alpha = 1/\sum_{i=0}^{N_f-1} A[i]$. To implement a stochastic weighted average using multiplexers, the weights are generated from the absolute values of the coefficients A[i] (i = 0, 1, …, N_f). Therefore, (9) can be changed to

$$Y = \alpha \sum_{i=0}^{N_f-1} |A[i]| \cdot \mathrm{sign}(A[i])\, X[n-i], \tag{10}$$

where $\alpha = 1/\sum_{i=0}^{N_f-1} |A[i]|$. To implement the weighted adder for an N_f-tap FIR filter, $\lceil \log_2 N_f \rceil$ selection signals are required. As an example, consider the design of a 4-tap FIR filter. In this case, A[0], A[1], A[2] and A[3] are the conventional binary coefficients determined by the filter specification. The sign of a coefficient is implemented using an XOR gate at the data input of the weighted adder. If the coefficient is positive, the XOR gate becomes transparent without changing the input value. If the coefficient is negative, the XOR gate acts like an inverter to invert the corresponding input value. The two selection signals are determined by the weights or filter coefficients, and they are used to select one of the four multiplexer inputs (see Fig. 8 (b)). The probability that each combination appears (e.g. Sel[0] = 0, Sel [1] = 1) is the normalized coefficient for that input (e.g.$|A[1]| \cdot \alpha$). The stochastic representation of a number must lie within the interval of [0, 1], so the coefficients are already normalized by the factor α in (9) and (10). The relationship between the specified coefficients and the probabilities of selecting the corresponding inputs (see Fig. 8) is given by

$$P\{Sel[0] = 0\} = (|A[0]| + |A[1]|) \cdot \alpha, \tag{11}$$

$$P\{Sel[0] = 1\} = 1 - (|A[0]| + |A[1]|) \cdot \alpha, \tag{12}$$

$$P\{Sel[1] = 0 \mid Sel[0] = 0\} = |A[0]|/(|A[0]| + |A[1]|), \tag{13}$$

$$P\{Sel[1] = 1 \mid Sel[0] = 0\} = |A[1]|/(|A[0]| + |A[1]|), \tag{14}$$

$$P\{Sel[1] = 0 \mid Sel[0] = 1\} = |A[2]|/(|A[2]| + |A[3]|), \tag{15}$$

$$P\{Sel[1] = 1 \mid Sel[0] = 1\} = |A[3]|/(|A[2]| + |A[3]|). \tag{16}$$

Here, P{Sel[X] = Y} denotes the probability that the select signal Sel[X] (X = 0 or 1) is Y (Y = 0 or 1). P{ $Sel[X_1] = Y_1 \mid Sel[X_2] = Y_2$ } denotes the probability of the select signal bit $Sel[X_1]$ (X_1 = 0 or 1) being Y_1 (Y_1 = 0 or 1) under the condition that the select signal bit $Sel[X_2]$ (X_2 = 0 or 1) is Y_2 (Y_2 = 0 or 1). Stochastic sequences can then be generated to represent the corresponding select signals. Equations (11) through (16) can be implemented using the Weight Generator (WG) in Fig. 8 (a). Then the weighted addition can be realized using the Weighted Adder (WA) in Fig. 8 (b) with the selecting signals provided by instances of the WG in Fig. 8 (a).

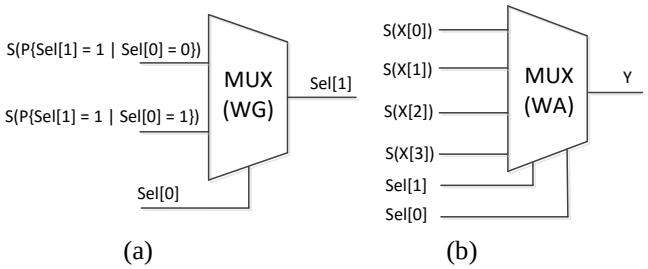


Fig. 8. A 4-term sum of products implemented using (a) a stochastic Weight Generator (WG) and (b) a stochastic Weighted Adder (WA).

The overall schematic of a 16-tap FIR filter using the proposed weighted average structure is shown in Fig. 9. Three multiplexers (WG1, WG2 and WG3) are needed as weight generators for the four selecting signals. For the selecting signals, Sel[0] comes directly from a SNG that encodes its corresponding binary value. The signal Sel[0] is a stochastic sequence encoding P{Sel[0] = 1} as shown in Equation (12). Sel[1] is the output of the 2:1 MUX whose selecting signal is Sel[0]. Signals Sel[2] and Sel[3] are generated similarly. Therefore we have four different sizes of multiplexers in the core of the stochastic FIR filter design. The 16:1 MUX (WA) implements the sum of products as a weighted average while the other three multiplexers implement weight generators. Note that the bipolar SNGs are used at the data inputs of the weighted adder, and unipolar SNGs are used elsewhere.

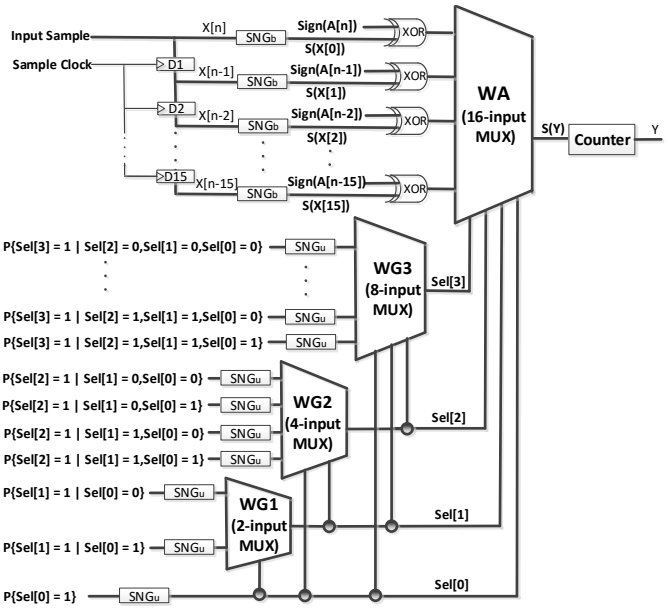


Fig. 9. The 16-tap FIR filter implemented with the multi-level weighted average (MWA) design.

Table 3. Performance of the FIR filters using the conventional binary approach, the stochastic MWA approach and the stochastic HWA approach. N_b: bit resolution, PR: passband ripple and SA: stopband attenuation.

Resolution (Bits)	Conventional Binary Implementation		Stochastic MWA Implementation				Stochastic HWA Implementation			
	PR (dB)	SA (dB)	N_s (bits)	PR (dB)	SA (dB)	PMM	N_s (bits)	PR (dB)	SA (dB)	PMM
3	3.0776	-3.9121	64	2.5327	-7.0345	8	64	2.5826	-7.1452	8
4	4.7882	-9.9327	256	1.3941	-11.3365	16	256	1.4147	-11.4942	16
5	2.8602	-13.1133	512	1.3016	-14.5083	16	512	1.3247	-14.7042	16
6	1.4623	-17.7985	1,024	0.7843	-17.6721	16	1,024	0.8017	-17.9031	16
7	0.7215	-24.0151	2,048	0.5391	-20.3922	16	2,048	0.5522	-20.6530	16
8	0.3913	-27.6340	8,192	0.2017	-27.2137	32	8,192	0.2098	-27.5353	32
9	0.1795	-31.1319	16,384	0.2835	-31.6085	32	16,384	0.2826	-31.9565	32
10	0.1026	-36.3678	65,536	0.0733	-35.6365	64	65,536	0.0715	-36.0254	64
11	0.0517	-45.5820	524,288	0.0488	-44.1428	256	524,288	0.0507	-44.6030	256
12	0.0324	-49.9427	1,048,576	0.0419	-47.8491	256	1,048,576	0.0373	-48.3225	256
13	0.0392	-51.4476	4,194,304	0.0342	-51.0634	512	4,194,304	0.0348	-51.5738	512
14	0.0350	-54.5942	16,777,216	0.0370	-54.9710	1,024	16,777,216	0.0369	-55.5106	1,024
15	0.0360	-55.9094	134,217,728	0.0327	-54.3603	4,096	134,217,728	0.0351	-54.8941	4,096
16	0.0364	-54.9557	536,870,912	0.0404	-54.7923	8,192	536,870,912	0.0364	-55.3241	8,192

IV. Performance Evaluation of the Conventional Binary and the Proposed Stochastic FIR Filters

A low-pass FIR filter design was considered to evaluate the proposed stochastic and binary filter designs. Detailed specifications of the low-pass filter are given in Table 2.

Table 2. Low-pass FIR filter specifications

Specifications	Values
Cut-off frequency f_c	100 Hz
The width of the transition band *BW*	30 Hz
Minimum stop-band attenuation	-50 dB
Maximum peak-to-peak pass-band ripple	0.1 dB

The filter was designed using a Hamming window, with the embedded Matlab function *fir1*(). The number of taps is 267, i.e., N_f = 267. The filter coefficients are obtained to meet the specifications in Table 2. The binary filter operates by converting the floating point numbers to fixed point numbers at different resolutions. The stochastic filters are built using the hard-wired weighted average (HWA) design and the multi-level weighted average (MWA) design. The sequence length N_s is given by $N_S = 2^{N_{LFSR}}$, where N_{LFSR} is the number of bits in an LFSR. Various sequence lengths were investigated for different resolutions to compare with the conventional binary design. In this experiment, N_{LFSR} was varied from 3 up to 30 (i.e, N_{LFSR} = 3, 4, …, 30) to determine the sequence length. The resolution N_b ranges from 3 bits to 16 bits. The magnitude responses of the filters were then investigated. Passband ripples (PRs) and stopband attenuations (SAs) are used to evaluate the performance of the binary design for various resolutions and the stochastic designs for different sequence lengths. Here PR and SA are defined as the maximum passband overshoot amplification and the minimum stopband attenuation, respectively. In this experiment, the passband, transition band and stopband are given as [0, 100 Hz], (100 Hz, 120 Hz] and (120 Hz, $+\infty$), respectively. The results are shown in Table 3.

In Table 3, the PRs and SAs are shown for different bit resolutions and sequence lengths for the MWA-based stochastic filter and the HWA-based stochastic filter. The stochastic FIR filters suffer from both quantization error and random fluctuations, but they show gradually-improving performance as the sequence length increases. For each bit resolution, the minimum sequence length N_s was found that matches the performance of the stochastic FIR filters to that of the conventional N_b-bit binary FIR filter. The performance matching multiplier (PMM) is then calculated by $PMM = N_s/2^{N_b}$. For example, for the 8-bit binary FIR filter, the HWA-based stochastic FIR filter using sequences with at least 8192 bits has smaller or similar passband ripples and stopband attenuations. The performance matching multiplier is thus $PMM = 8192/2^8 = 32$. Therefore, the HWA-based stochastic implementation using 8192-bit sequences is considered as the best case to compare with the 8-bit binary conventional implementation. The same strategy was applied to determine the proper sequence lengths for various resolutions for the MWA-based stochastic filter.

Table 4. Root mean square error (RMSE) comparison of the FIR filters using the conventional binary (CB) approach, the stochastic MWA approach and the stochastic HWA approach.

Resolution (bits)	RMSE (%)			Stochastic Sequence Length (bits)
	CB	MWA	HWA	
3	6.533	6.219	6.028	64
4	3.158	3.030	3.216	256
5	1.626	1.598	1.582	512
6	0.795	0.772	0.811	1,024
7	0.392	0.381	0.377	2,048
8	0.192	0.184	0.177	8,192
9	0.092	0.102	0.093	16,384
10	0.051	0.051	0.049	65,536
11	0.025	0.027	0.023	524,288
12	0.012	0.014	0.012	1,048,576
13	0.006	0.005	0.006	4,194,304
14	0.006	0.006	0.005	16,777,216
15	0.005	0.004	0.006	134,217,728
16	0.005	0.005	0.006	536,870,912

Table 5. Comparison of hardware cost, power consumption and minimum clock period for conventional binary (CB), CWA-based, MWA-based and HWA-based stochastic FIR filters including all the auxiliary circuits such as SNGs and counters.

Resolution (Bits)	Area (μm^2)				Power (mW)				Min Clock Period (ps)			
	CB	CWA	MWA	HWA	CB	CWA	MWA	HWA	CB	CWA	MWA	HWA
3	12,757	14,944	12,972	12,855	24.24	20.35	19.89	19.82	390	370	310	340
4	21,262	17,415	15,815	15,525	35.50	29.79	29.35	29.05	410	370	310	350
5	31,893	21,023	18,948	18,752	47.86	40.17	38.21	38.02	420	400	320	360
6	44,650	26,841	22,331	21,998	61.22	51.39	47.56	47.96	440	400	320	360
7	59,533	27,989	24,801	24,710	75.49	63.36	59.31	59.06	470	410	340	390
8	76,543	27,690	27,782	27,410	90.59	76.04	71.68	71.55	490	410	360	400
9	95,679	33,545	30,796	29,986	106.49	89.38	84.82	84.66	490	420	380	410
10	116,943	36,248	33,655	33,122	123.12	103.3	98.05	97.23	500	430	380	410
11	140,330	38,343	36,690	36,510	140.45	117.9	109.2	109.0	510	470	390	420
12	165,843	44,999	43,761	43,536	138.45	133.1	127.5	125.2	540	470	400	440
13	193,482	46,574	45,921	44,910	158.45	148.7	141.5	140.7	550	470	400	440
14	223,269	50,162	48,774	48,067	177.10	164.8	159.8	159.2	570	490	410	450
15	255,141	53,746	51,991	50,829	196.36	181.5	176.1	174.8	590	490	420	460
16	289,160	57,322	55,039	54,898	226.64	198.6	192.8	189.6	640	500	440	460

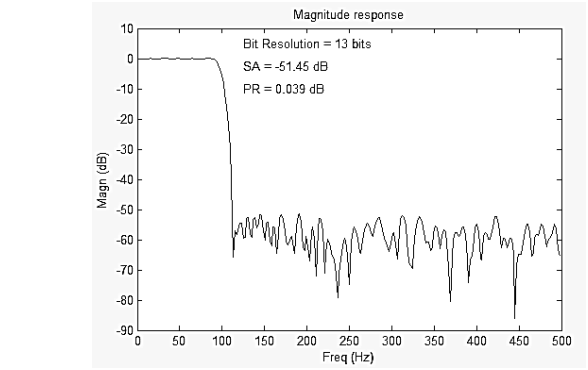
To further show the performance of the proposed designs, results on the root mean square error (RMSE) are provided in addition to PR and SA (see Table 4). The RMSE is given by

$$NMSE = \sqrt{\frac{1}{N_{trial}}\sum_{i=0}^{N_{trial}-1}(Y_i - Y_i')^2}, \qquad (17)$$

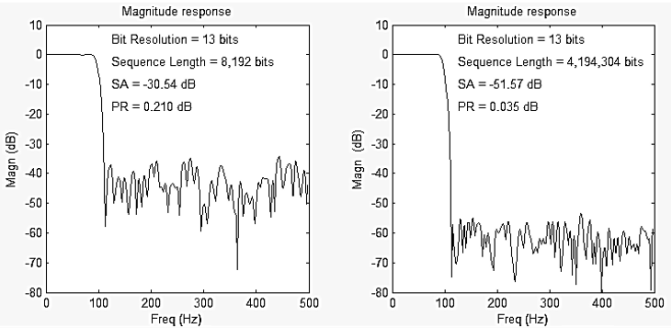
where Y_i and Y_i' are the expected correct output and the actual output of the FIR filter, respectively, and N_{trial} is the number of trials or simulations. RMSE indicates the improvement in accuracy from using higher resolutions and longer sequences.

The minimum resolution to achieve the filter specifications in Table 2 is 13 bits for the binary design. The attenuation in the stopband is -51.4476 dB and the passband ripple is 0.0392 dB. The magnitude responses of the stochastic and binary filters are plotted in Fig. 10 for 13-bit resolution. The HWA-based stochastic design with the minimum sequence length suffers from a maximum passband ripple of 0.2098 dB, as shown in Fig. 10 (b). The stopband attenuation for the HWA-based FIR filter is only -30.5392 dB (Fig. 10 (b)). To match the performance of the binary filter, the PMM has to be increased to 512 (Fig. 10 (c)). The PR and SA are 0.0348 dB and -51.5738 dB, respectively, for the HWA-based stochastic filter in this case. We can draw the same conclusion for the MWA-based FIR filter in Table 3.

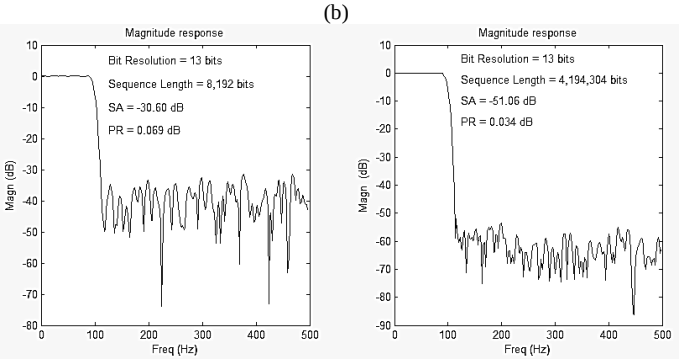
These results are consistent with the theory in [11] that a rather long stochastic bit stream is required to achieve the same root mean square error (RMSE). The necessary stochastic bit stream length is found to be $L=2^{2n+1}$ (n is the binary bit resolution) due to the intrinsic fluctuation errors and the signal to noise ratio (SNR) decrease caused by multiple stages [11]. Such a long stochastic sequence means a relatively large computing time. Note that using variance reduction techniques such as non-Bernoulli sequences can increase accuracy and thus decrease PMM [12]. However, this comes at the price of requiring more hardware. For instance, non-Bernoulli sequence generation requires a process where a certain number of 1's and 0's are generated and then scrambled by a pseudo-random number generator to achieve randomness. The area required by the stochastic number generators is often more problematic compared to the hardware required by the core



(a)



(b)



(c)

Fig. 10. Magnitude responses of 13-bit FIR filters: (a) Conventional binary design, (b) Stochastic HWA design without and with performance matching (*PMM* = 512) , (c) Stochastic MWA design without and with performance matching (*PMM* = 512).

Table 6. Comparison of hardware cost, power consumption and minimum clock period for conventional binary (CB), CWA-based, MWA-based and HWA-based stochastic FIR filters without any auxiliary circuits such as SNGs and counters.

Resolution (Bits)	Area (μm^2)				Power (mW)				Min Clock Period (ps)			
	CB	CWA	MWA	HWA	CB	CWA	MWA	HWA	CB	CWA	MWA	HWA
3	12,757	8,719	8,747	2,565	24.24	15.7	15.5	4.0	390	220	210	170
4	21,262	8,719	8,747	3,096	35.5	15.7	15.5	5.8	410	220	210	180
5	31,893	8,719	8,747	3,749	47.86	15.7	15.5	6.6	420	220	210	180
6	44,650	8,719	8,747	4,399	61.22	15.7	15.5	7.5	440	220	210	180
7	59,533	8,719	8,747	4,931	75.49	15.7	15.5	9.8	470	220	210	200
8	76,543	8,719	8,747	5,461	90.59	15.7	15.5	11.3	490	220	210	200
9	95,679	8,719	8,747	5,980	106.49	15.7	15.5	12.9	490	220	210	210
10	116,943	8,719	8,747	6,620	123.12	15.7	15.5	14.4	500	220	210	210
11	140,330	8,719	8,747	7,287	140.45	15.7	15.5	15.7	510	220	210	210
12	165,843	8,719	8,747	8,669	138.45	15.7	15.5	17.0	540	220	210	220
13	193,482	8,719	8,747	8,947	158.45	15.7	15.5	18.1	550	220	210	220
14	223,269	8,719	8,747	9,602	177.1	15.7	15.5	19.8	570	220	210	230
15	255,141	8,719	8,747	10,155	196.36	15.7	15.5	20.8	590	220	210	230
16	289,160	8,719	8,747	10,964	226.64	15.7	15.5	21.3	640	220	210	230

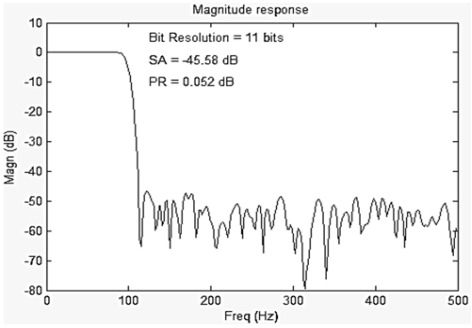
circuit in stochastic computing. Therefore, any variation reduction technique should be carefully evaluated to determine if the advantage would justify the increased overhead.

Note that a shorter sequence length can be used for the HWA-based stochastic filter to obtain a degraded but possibly still acceptable performance. For example, if the PMM is 32 instead of 512, the SA becomes -42.3512 dB and the PR is 0.0593 dB (see Fig. 11 (b)). However, the time and energy per computed output is reduced to only 1/16 of the previous result. Similarly, the MWA-based stochastic FIR filter also benefits from the property of graceful degradation in performance (see Fig. 11 (c)). In contrast, an 11-bit conventional binary implementation of the filter shows similarly degraded performances with an SA of -45.5820 dB and a PR of 0.0517 dB (see Fig. 11 (a)), however with very little saving in energy consumption. This is discussed in more detail next.

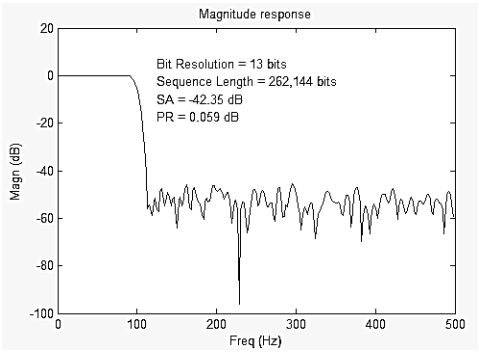
V. Simulation Results

For hardware performance comparison, the Synopsys Design Compiler was used to synthesize a high-level design in VHDL into a standard cell ASIC design. Metrics such as silicon area, power consumption and delay were obtained.

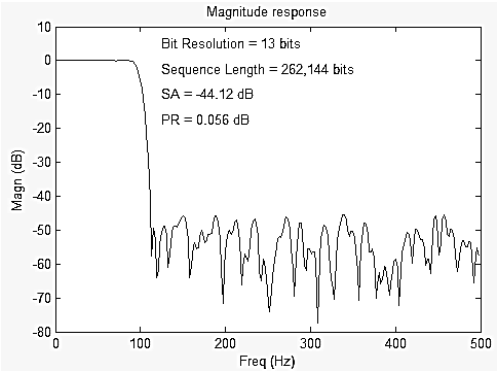
The FIR filters specified in Table 2 were simulated for various resolutions from 3 bits to 16 bits. In Table 5, the circuit performance is compared with respect to silicon area, power consumption and delay. Although the core of the stochastic circuit is implemented using XNOR gates and multiplexers as adders and multipliers, the interfacing circuits require a relatively large number of SNGs and counters, especially for FIR filters with a large number of taps. The hardware cost of binary circuits grows faster than that of the stochastic circuits, so for a larger resolution, the stochastic circuits become increasingly advantageous over a binary design. The auxiliary circuits such as the SNGs and counters, however, make this advantage of stochastic circuits less significant. Although stochastic logic gates such as multiplexers and XNOR gates are inexpensive, the auxiliary circuits used for conversions can be costly in terms of silicon area and power. In fact, in a 12-bit HWA-based stochastic FIR filter, we found that 80.3% of the silicon area comes from the stochastic number generators and counters.



(a)



(b)



(c)

Fig. 11. Magnitude responses of lower-quality FIR filters: (a) 11-bit conventional binary design, (b) 13-bit stochastic HWA design with *PMM* = 32, (c) 13-bit stochastic MWA design with *PMM* = 32.

Note that both the binary and stochastic circuits have been optimized for maximum throughput by adding pipeline registers as determined by the Synopsys synthesis tool. The area could be over-estimated for this reason. As shown in Table 3, the sequence length explains the long latency required in the operation of a stochastic circuit. Adopting a faster clock is a potential way to reduce latency. With the help of timing analysis, the clock can be pushed to the limit according to the slack time. The results are shown in Table 5. The required stochastic sequence length is given by $2^{N_b} \cdot PMM$, where N_b (N_b = 3, 4, …, 16) is the binary resolution and *PMM* is the performance matching multiplier. The reported power consumptions are estimated at the fastest clocks for each of the resolutions.

Techniques have been proposed that avoid the auxiliary circuits such as SNGs. For example, the authors in [9] proposed to generate random bits from analog signals using sigma-delta modulation. We therefore decided to present the circuit performance when auxiliary circuits are excluded. Only the core of the stochastic circuits is considered without the auxiliary circuits such as SNGs and counters. Area, power and minimum clock period results are reported in Table 6. The core of the CWA-based and MWA-based circuits for the 267-tap FIR filter does not change for various bit resolutions. The HWA-based implementation becomes more complex as the bit resolution increases. This is because the HWA-based structure depends on the coefficients of the filter. In Table 6, it can be seen that the stochastic circuits use less hardware area and consume less power compared to the conventional binary implementation as expected. The advantage of stochastic circuits becomes more significant as the bit resolution increases.

It can be seen that the stochastic circuits are more compact and they consume less energy per clock cycle than conventional implementations. However, they suffer from the long latency caused by the required stochastic sequences, which makes the total energy per operation (EPO) less competitive. EPO is obtained as the product of power and the time required for performing one operation. Throughput per area (TPA) is further considered as the number of operations per circuit area in a unit time. When computing the TPA and the EPO, the stochastic FIR filter must work as effectively as the binary conventional FIR filter. The effectiveness is measured using the performance metrics passband ripple (PR) and stopband attenuation (SA) in Table 3, so the sequence lengths in Table 3 must be used, which makes the results even less competitive at high resolutions. In fact, the stochastic approach is no longer competitive in terms of TPA and EPO when long sequences have to be used in a stochastic implementation.

When the auxiliary circuits, such as stochastic number generators and counters, are shared in a large circuit, their cost may be acceptably small compared to the core stochastic circuit. As the two proposed stochastic designs have similar performance, we use the stochastic HWA-based circuit to compare with the conventional binary circuit. Figs. 12 and 13 show plots of EPO and TPA, respectively, for the binary and stochastic HWA-based circuits (with and without the auxiliary circuits). For stochastic implementations, the sequence length is the factor that dominates the overall circuit performance in terms of TPA and EPO. As the required sequence lengths are the same for both the HWA-based and MWA-based implementations, the circuit performances of the two designs are also similar. Therefore we only show the TPA and EPO for the stochastic HWA-based circuit. The *y*-axes in both plots are the base-10 logarithms of the original metrics. The *x*-axes are the bit resolutions from 3 bits to 16 bits. The two figures show that the stochastic approach is not competitive in terms of the EPO and TPA. The higher the resolution, the less competitive the stochastic implementation becomes. This is caused by the required sequence length, which grows exponentially with the bit resolution. When the auxiliary circuits are not considered, the stochastic circuit shows a better performance. In particular, it performs better in terms of the TPA than the binary design for resolutions below 5 bits.

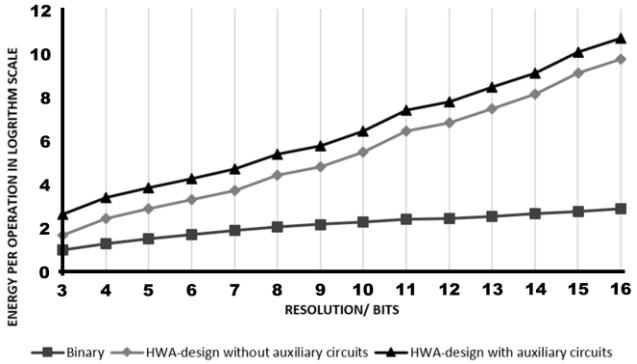


Fig. 12. Energy per Operation comparison: Stochastic HWA Design (with/without auxiliary circuits) and the Binary Design.

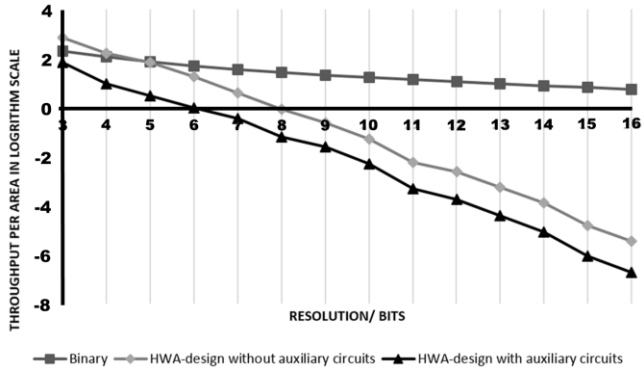


Fig. 13. Throughput per Area comparison: Stochastic HWA Design (with/without auxiliary circuits) and the Binary Design.

Although the stochastic designs suffer from long latencies, their performance degrades gracefully as the energy is reduced. Take the 13-bit designs as an example. As shown in Table 7, a lower-quality HWA-based stochastic filter is implemented using 262,144 bits (compared to 4,194,304 bits required by an HWA-based filter that matches the performance of a 13-bit conventional binary filter). An 11-bit conventional binary filter shows similarly degraded performance. The stopband attenuation of both the lower-quality filters is 6 dB higher than that of the good-quality filters. As the HWA-based stochastic filter only requires 1/16 of the original sequence length, the EPO is just 6.25% of that for the good-quality HWA-based filter. Similarly, the EPO of the lower quality MWA-based stochastic filter is only 6.32% of that for the good-quality

MWA-based filter. However, the 11-bit binary filter consumes 82.19 % of the energy per operation compared to the 13-bit binary filter. The EPOs of these lower-quality filters are shown in Table 7.

Table 7. Energy savings with lower-quality implementations for conventional binary (CB), MWA-based and HWA-based FIR filters

Implementations	CB	MWA	HWA
EPO of Higher Quality Filter (*pJ*)	87.15	22,397,583,360	253,700,027
EPO of Lower Quality Filter (*pJ*)	71.63	1,415,848,960	15,856,251
Energy Saving (%)	17.81	93.68	93.75

VI. Error Analysis

A. Sources of Errors and Inaccuracies

In conventional FIR filters, errors are mainly due to the finite word length effect. Because of the limited size of registers, a real number has to be rounded to the nearest number that the computer can store. More specifically, the input quantization, coefficient quantization and truncated products in multiplication are the major sources of such inaccuracies. In stochastic circuits, errors are also caused by the random fluctuation of the stochastic bits. This type of error or noise is propagated through the computation process to the circuit output.

B. Error Analysis for Conventional Binary Filters

1) Quantization errors due to the finite word length effect

When a signal is quantized at the inputs of an FIR filter, both the rounding and floor functions are applied. Because the floor function can introduce a biased error distribution into the digital system, we only consider rounding in our experiments. The quantization error caused by the finite word length effects is then considered. For any digital system with N_b-bit resolution, an N_b'-bit normalized binary number x in [0, 1] has to be rounded off to become a normalized binary number x' with N_b bits. By normalized we mean that the weights of bits (from the most significant bit to the least significant bit) in an N_b-bit binary number form a geometric series $\{2^{-1}, 2^{-2}, \dots, 2^{-N_b}\}$. The weight a_i of the i^{th} binary bit is either 0 or 1. The values of x and x' are therefore between 0 and 1:

$$x = \sum_{i=1}^{N_b'} a_i \cdot 2^{-i} = \sum_{i=1}^{N_b} a_i \cdot 2^{-i} + \sum_{i=N_b+1}^{N_b'} a_i \cdot 2^{-i};$$
$$x' = \begin{cases} \sum_{i=1}^{N_b} a_i \cdot 2^{-i}, \text{if } a_{N_b+1} = 0, \\ 2^{-N_b} + \sum_{i=1}^{N_b} a_i \cdot 2^{-i}, \text{if } a_{N_b+1} = 1. \end{cases} \quad (18)$$

The quantization error e_q is calculated as

$$e_q = x - x' =$$
$$\begin{cases} \sum_{i=N_b+2}^{N_b'} a_i \cdot 2^{-i}, \text{if } a_{N_b+1} = 0, \\ -2^{-(N_b+1)} + \sum_{i=N_b+2}^{N_b'} a_i \cdot 2^{-i}, \text{if } a_{N_b+1} = 1. \end{cases} \quad (19)$$

Let Δ be the minimum distance between two numbers represented by N_b bits, i.e.,

$$\Delta = 2^{-N_b}. \quad (20)$$

Then the quantization error e_q is bounded by

$$-\frac{\Delta}{2} \le e_q < \frac{\Delta}{2}. \quad (21)$$

The quantization error e_q is usually defined as a white noise which is independent of the input signal. Therefore, we establish a statistical model for the quantization error as described in Fig. 14. The quantization error e_q is an additive variable to the linear system. Among all possible values, e_q is considered evenly distributed. The probability density function (PDF) of the quantization error e_q is described as [14].

$$f(e_q) = \begin{cases} \frac{1}{\Delta}, -\frac{\Delta}{2} \le e_q < \frac{\Delta}{2}; \\ 0, \quad others. \end{cases} \quad (22)$$

Based on the PDF, the mean and the variance of e_q are given by

$$E(e_q) = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} e_q \cdot f(e_q) de_q = 0. \quad (23)$$

$$\sigma_q^2 = E\left[(e_q - E(e_q))^2\right] = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} (e_q - E(e_q))^2 \cdot f(e_q) de_q = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} \frac{1}{\Delta} e_q^2 de_q = \frac{\Delta^2}{12} = \frac{1}{12 \cdot 2^{2N_b}}. \quad (24)$$

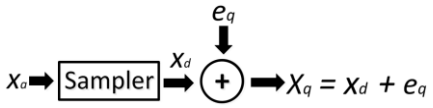


Fig. 14. A statistical model for the quantization error e_q: X_a is the analog input and X_d is the digital sample from X_a; X_q is the quantized sample with quantization error e_q [14].

2) Inaccuracies in the computation of conventional FIR filters

A statistical model is developed to evaluate error accumulation and propagation in the computation. Because an FIR filter is a linear system, errors added to the FIR filters are propagated to the next arithmetic operation with similar characteristics. To start the analysis, the original/theoretical FIR filter in (6) is simplified to

$$Y = \sum_{i=0}^{N_f-1} H_i X_i, \quad (25)$$

where H_i and X_i are numbers with infinite precision. By quantization, the actual result Y' is biased from the expected result Y, i.e.,

$$Y' = \sum_{i=0}^{N_f-1} (H_i + e_{Hi})(X_i + e_{Xi})$$
$$= \sum_{i=0}^{N_f-1} (H_i X_i + X_i e_{Hi} + H_i e_{Xi} + e_{Hi} e_{Xi}) \quad (26)$$
$$= Y + \sum_{i=0}^{N_f-1} (X_i e_{Hi} + H_i e_{Xi} + e_{Hi} e_{Xi}),$$

where e_{Hi} and e_{Xi} are quantization errors of the coefficient H_i and the input X_i. Both e_{Hi} and e_{Xi} can be modeled by e_q in (19). To calculate the mean $E(Y')$, i.e.,

$$E(Y') = E(Y) + E\left(\sum_{i=0}^{N_f-1} (X_i e_{Hi} + H_i e_{Xi} + e_{Hi} e_{Xi})\right), \quad (27)$$

we first determine the mean of the cross product term $e_{Hi} e_{Xi}$. Due to [15], if e_{Hi} and e_{Xi} are independent real-valued continuous random variables with finite expected values, then

$$E(e_{Hi} e_{Xi}) = E(e_{Hi}) E(e_{Xi}) = 0. \quad (28)$$

Taking into consideration (23), (27) and (28), we obtain

$$E(Y') - E(Y) = \sum_{i=0}^{N_f-1} (X_i E(e_{Hi}) + H_i E(e_{Xi}) + E(e_{Hi} e_{Xi})) = 0. \quad (29)$$

By definition and (29), the variance of the output Y' is

$$Var(Y') = E\left[(Y' - E(Y'))^2\right] = E\left[(Y' - Y)^2\right]. \quad (30)$$

On the other hand, the variance of Y' is also given by

$$Var(Y') = \sum_{i=0}^{N_f-1} (X_i^2 Var(e_{Hi}) + H_i^2 Var(e_{Xi}) + Var(e_{Hi} e_{Xi})) = \sum_{i=0}^{N_f-1} (X_i^2 \sigma_q^2 + H_i^2 \sigma_q^2 + Var(e_{Hi} e_{Xi})). \quad (31)$$

Because $Var(e_{Hi} e_{Xi})$ is of a higher order than the other terms in (31), we have

$$\max\{Var(e_{Hi}e_{Xi})\} = \frac{\Delta^4}{16} \ll \sum_{i=0}^{N_f-1} \sigma_q^2(X_i^2 + H_i^2) = \sum_{i=0}^{N_f-1} \frac{\Delta^2}{12}(X_i^2 + H_i^2). \quad (32)$$

Therefore, $Var(e_{Hi}e_{Xi})$ in (31) can be ignored and by (24) we obtain

$$Var(Y') \approx \sum_{i=0}^{N_f-1} \sigma_q^2(X_i^2 + H_i^2) = \frac{\Delta^2}{12}\sum_{i=0}^{N_f-1}(X_i^2 + H_i^2). \quad (33)$$

From (33), it can be seen that the quantization error is amplified by a multiplicative factor. Then, e_c is defined as the overall error for the conventional binary filter, i.e.,

$$e_c = |Y' - Y| \quad (34)$$

By combining (30), (33) and (34), we obtain the mean of the squared error as

$$E(e_c^2) = E((Y' - Y)^2) = Var(Y') = \frac{\Delta^2}{12}\sum_{i=0}^{N_f-1}(X_i^2 + H_i^2) = \frac{1}{12\cdot 2^{2N_b}}\sum_{i=0}^{N_f-1}(X_i^2 + H_i^2) \quad (35)$$

By (35), e_c can be estimated as

$$e_c = |Y' - Y| \approx \sqrt{\frac{\Delta^2}{12}\sum_{i=0}^{N_f-1}(X_i^2 + H_i^2)} = \frac{\Delta}{2\sqrt{3}}\sqrt{\sum_{i=0}^{N_f-1}(X_i^2 + H_i^2)} = \frac{\sqrt{3}}{3\cdot 2^{N_b+1}}\sqrt{\sum_{i=0}^{N_f-1}(X_i^2 + H_i^2)}. \quad (36)$$

Hence, the computation error for the conventional binary implementation is related to the bit resolution N_b as well as the coefficients and inputs.

C. Error Analysis for Stochastic Filters

1) Quantization errors due to the finite sequence length effect

When a signal is stochastically encoded, the precision is limited by the sequence length N_s. For any N_s-bit sequence used in a stochastic system, the minimum distance between two numbers is

$$\Delta_s = \frac{1}{N_s}. \quad (37)$$

Then the quantization error is bounded by

$$-\frac{\Delta_s}{2} < e_{qs} \le \frac{\Delta_s}{2}. \quad (38)$$

Similarly to the conventional binary quantization error given in (19), the mean and variance of the stochastic quantization error are respectively

$$E(e_{qs}) = 0, \quad (39)$$

$$\sigma_{qs}^2 = \frac{{\Delta_s}^2}{12} = \frac{1}{12\cdot {N_s}^2}. \quad (40)$$

2) Quantization effect in stochastic computing

The propagation effect of quantization errors in stochastic computing is similar to that in conventional FIR filters. The effect of quantization errors in stochastic computing can be analyzed by evaluating the output $P_{Y'}$. If P_Y is the correct output without errors, we have

$$P_Y = \sum_{i=0}^{N_f-1} P_{H_i}P_{X_i}. \quad (41)$$

We further include the quantization effect by adding errors $P_{e_{H_i}}$, $P_{e_{X_i}}$ and P_{e_Y} to input P_{X_i}, the coefficient P_{H_i} and the output P_Y, respectively, where $i = 0, 1, \dots, N_f - 1$. Hence the output $P_{Y'}$ can be calculated by

$$\begin{aligned} P_{Y'} &= \sum_{i=0}^{N_f-1}(P_{H_i} + P_{e_{H_i}})(P_{X_i} + P_{e_{X_i}}) \\ &= \sum_{i=0}^{N_f-1}(P_{H_i}P_{X_i} + P_{X_i}P_{e_{H_i}} + P_{H_i}P_{e_{X_i}} + P_{e_{H_i}}P_{e_{X_i}}) \\ &= P_Y + \sum_{i=0}^{N_f-1}(P_{X_i}P_{e_{H_i}} + P_{H_i}P_{e_{X_i}} + P_{e_{H_i}}P_{e_{X_i}}). \end{aligned} \quad (42)$$

$P_{e_{H_i}}$ and $P_{e_{X_i}}$ are independent quantization errors that can be modeled by e_{qs} with the properties in (38), (39) and (40). The mean of the stochastic output $P_{Y'}$ is thus given by

$$E(P_{Y'}) = E(P_Y) + E\left(\sum_{i=0}^{N_f-1}\left(P_{X_i}P_{e_{H_i}} + P_{H_i}P_{e_{X_i}} + P_{e_{H_i}}P_{e_{X_i}}\right)\right) = E(P_Y). \quad (43)$$

The variance of the output $P_{Y'}$ can be obtained from (42) as

$$Var(P_{Y'}) = \sum_{i=0}^{N_f-1}\left(P_{Xi}^2 Var(P_{e_{H_i}}) + P_{Hi}^2 Var(P_{e_{X_i}}) + Var(P_{e_{H_i}}P_{e_{X_i}})\right) = \sum_{i=0}^{N_f-1}\left(P_{Xi}^2 + P_{Hi}^2\right)\sigma_{qs}^2 + Var(P_{e_{H_i}}P_{e_{X_i}})\Big). \quad (44)$$

$Var\left(P_{e_{H_i}}P_{e_{X_i}}\right)$ in (44) can be ignored due to the fact that

$$\max\left\{Var\left(P_{e_{H_i}}P_{e_{X_i}}\right)\right\} = \frac{\Delta_s^4}{16} \ll \sum_{i=0}^{N_f-1}\sigma_{qs}^2(P_{Xi}^2 + P_{Hi}^2) = \sum_{i=0}^{N_f-1}\frac{\Delta_s^2}{12}(P_{Xi}^2 + P_{Hi}^2). \quad (45)$$

Therefore, (46) becomes

$$Var(P_{Y'}) \approx \sigma_{qs}^2\sum_{i=0}^{N_f-1}(P_{Xi}^2 + P_{Hi}^2) = \frac{\Delta_s^2}{12}\sum_{i=0}^{N_f-1}(P_{Xi}^2 + P_{Hi}^2)\,. \quad (46)$$

By definition and (43), we have

$$Var(P_{Y'}) = E\left[\left(P_{Y'} - E(P_{Y'})\right)^2\right] = E[(P_{Y'} - P_Y)^2]. \quad (47)$$

By combining (46) and (47), we get

$$E[(P_{Y'} - P_Y)^2] = \frac{\Delta_s^2}{12}\sum_{i=0}^{N_f-1}(P_{Xi}^2 + P_{Hi}^2)\,. \quad (48)$$

The overall quantization error e_{oq} caused by the quantization effect can be defined as the absolute difference between the calculated result $P_{Y'}$ and the accurate result P_Y, i.e.,

$$e_{oq} = |P_{Y'} - P_Y|. \quad (49)$$

From (48) and (49), the mean of the squared error e_{oq}^2 can be evaluated as

$$E\left[e_{oq}^2\right] = E[(P_{Y'} - P_Y)^2] = \frac{\Delta_s^2}{12}\sum_{i=0}^{N_f-1}(P_{Xi}^2 + P_{Hi}^2)\,. \quad (50)$$

Because of (37) and (50), e_{oq} can be approximated as

$$e_{oq} \approx \sqrt{\frac{\Delta_s^2}{12}\sum_{i=0}^{N_f-1}\left(P_{Xi}^2 + P_{Hi}^2\right)} = \frac{\sqrt{3}}{6N_s}\sqrt{\sum_{i=0}^{N_f-1}\left(P_{Xi}^2 + P_{Hi}^2\right)}\,. \quad (51)$$

3) Random fluctuations in stochastic computing

In addition to the quantization error caused by the limited sequence length, stochastic computing also suffers from a fluctuation error as the pseudo random number generator introduces uncertainty into the stochastic representations. The fluctuation error is denoted by e_{fs}. $P_{Y''}$ is defined as the final result obtained by stochastic computing [11, 16]. Due to random fluctuations, $P_{Y''}$ is different from P_Y. If P_Y is encoded by the stochastic bit stream $y(i)$, where $i = 1, 2, \dots, N_s$ and N_s is the sequence length, then the final output $P_{Y''}$ is obtained by

$$P_{Y''} = \frac{1}{N_s}\sum_{i=1}^{N_s} y(i), \quad (52)$$

If no stochastic fluctuations existed, there is no difference between P_Y and $P_{Y''}$. However, the stochastic bit stream $y(i)$ is usually a Bernoulli sequence. The mean of the output $P_{Y''}$ is

$$E(P_{Y''}) = P_Y. \quad (53)$$

The variance is given by

$$Var[P_{Y''}] = Var\left[\frac{1}{N_s}\sum_{i=1}^{N_s} y(i)\right] = E\left[\left(P_{Y''} - E(P_{Y''})\right)^2\right] = E[(P_{Y''} - P_Y)^2]. \quad (54)$$

As $y(i)$ (i = 1, 2, …, N_s) is a Bernoulli sequence, we have

$$Var\left[\frac{1}{N_s}\sum_{i=1}^{N_s} y(i)\right] = \frac{P_Y(1-P_Y)}{N_s}. \quad (55)$$

Let e_{fs} be the fluctuation error defined as

$$e_{fs} = |P_{Y''} - P_Y|. \quad (56)$$

The error is measured using the variance in (54) and (55), i.e.,

$$E[e_{fs}^2] = E[(P_{Y''} - P_Y)^2] = \frac{P_Y(1-P_Y)}{N_s}. \quad (57)$$

Therefore the fluctuation error can be approximated as

$$e_{fs} = \sqrt{\frac{P_Y(1-P_Y)}{N_s}}. \quad (58)$$

Equation (58) implies that the fluctuation can be controlled by simply using longer sequences.

4) Inaccuracies in the computation of stochastic FIR filters

The stochastic computation can suffer from both quantization errors and fluctuation errors. The overall error e_{os} in a stochastic filter is thus the sum of the two errors:

$$e_{os} = e_{oq} + e_{fs}. \quad (59)$$

Substituting the quantization error e_{oq} in (51) and the fluctuation error e_{fs} in (58), we have

$$e_{os} = \sqrt{\frac{P_Y(1-P_Y)}{N_s}} + \frac{\sqrt{3}}{6N_s}\sqrt{\sum_{i=0}^{N_f-1}(P_{Xi}^2 + P_{Hi}^2)}, \quad (60)$$

where P_{X_i}, P_{H_i} and P_Y in (60) are coefficients, inputs and outputs, respectively, for the stochastic filter.

D. Stochastic Sequence Length Estimate by Error Analysis

The stochastic sequence length is an important parameter as it determines both the computational accuracy and the circuit performance. To determine the sequence length for stochastic FIR filters, we use (36) to evaluate the overall error of the conventional binary circuit. For the stochastic circuit, we use the upper bound of the overall stochastic error e_{os} in (60) as an approximation of the stochastic error. To understand the relationship between bit resolution N_b and sequence length N_s, let $e_{os} = e_c$, i.e.,

$$e_{os} \approx \sqrt{\frac{P_Y(1-P_Y)}{N_s}} + \frac{\sqrt{3}}{6N_s}\sqrt{\sum_{i=0}^{N_f-1}(P_{Xi}^2 + P_{Hi}^2)} = \frac{\sqrt{3}}{6\cdot 2^{N_b}}\sqrt{\sum_{i=0}^{N_f-1}(X_i^2 + H_i^2)} \approx e_c. \quad (61)$$

Hence the stochastic sequence length N_s is approximately

$$N_s \approx \frac{12P_Y(1-P_Y)}{\sum_{i=0}^{N_f-1}(X_i^2+H_i^2)} \cdot 2^{2N_b}. \quad (62)$$

To describe how N_s changes as N_b increases, we use the big O notation to estimate the relationship between the necessary stochastic sequence lengths for a stochastic FIR filter to match the performance of the conventional binary FIR filter. Therefore, (62) can be written as

$$N_s \approx O(2^{2N_b}). \quad (63)$$

(63) shows that the sequence length N_s grows exponentially as the binary resolution N_b increases. This is consistent with the stochastic sequences required in our experiments (see Table 3). Clearly, this fact has an adverse effect to any stochastic implementation. However, if a degraded performance in accuracy is acceptable, an exponential reduction would result in the required sequence length. This reduction in sequence length subsequently means a substantial reduction in energy consumption, thus achieving a significant improvement in performance metrics such as the EPO. For example, if the performance of a stochastic filter equivalent to that of an 8-bit binary filter is acceptable for an ideal 12-bit filter, the required sequence length, as per (63), would be only 1/256 of the length required for matching the performance of the 12-bit filter. This would reduce the EPO of the stochastic circuit by a factor of 255/256. However, the conventional binary implementation would only result in a much smaller energy reduction, as shown in Table 7 (albeit for a different example).

VII. Fault Tolerance Analysis and Simulation

Although errors are inevitable in the quantization process or caused by the inherent random fluctuation, stochastic computing has been known to be intrinsically fault-tolerant. In this section, we consider the fault tolerance of both the conventional binary and stochastic designs by taking into account soft errors.

A. Fault-tolerance Analysis

We first discuss the different behaviors of the conventional binary and stochastic circuits using the bit-flip error model [4, 17].

Consider a normalized N_b-bit binary number with value B:

$$B = x_1 \cdot 2^{-1} + x_2 \cdot 2^{-2} + \cdots + x_{N_b} \cdot 2^{-N_b}, \quad (64)$$

where x_i is the bit with weight 2^{-i} (i = 1, …, N_b). Let R_i be a random variable to indicate if an error occurs or not, i.e., if R_i is 1, an error occurs, so bit i flips. Further let ε be the error probability, i.e.,

$$P(R_i = 1) = \varepsilon. \quad (65)$$

Affected by possible bit flips, the normalized binary number becomes

$$B' = \sum_{i=1}^{N_b} x'_i \cdot 2^{-i} = \sum_{i=1}^{N_b}[R_i(1-x_i) + x_i(1-R_i)] \cdot 2^{-i}. \quad (66)$$

The error for the conventional binary approach is defined as

$$e_c^{(i)} = B' - B, \quad (67)$$

where a superscript (i) is used to indicate that $e_c^{(i)}$ is to denote the error caused by error injection. It has been shown in [4] that the error $e_c^{(i)}$ has the following mean value and variance:

$$E[e_c^{(i)}] \approx (1-2B)\varepsilon; \quad (68)$$

$$Var[e_c^{(i)}] \approx \frac{1}{3}(1-\varepsilon)\varepsilon. \quad (69)$$

Since the stochastic implementation does not necessarily use the minimum length for an N_b-bit binary number, the mean value and variance for the stochastic method are re-computed for comparison. The stochastic number S is the encoded value of the N_b-bit binary number B. As a performance matching multiplier (PMM) is used, the actual sequence length is $N_s = PMM \cdot 2^{N_b}$. For implementation convenience usually PMM is also a number in the form of $2^{N_b'}$ (N_b'= 0, 1, 2, …). Hence the N_s-bit stochastic sequence can be generated using a (N_b+N_b')-bit linear feedback shift register (LFSR). A stochastic bit stream $y_1 y_2 \ldots y_{N_s}$ is produced to encode a normalized N_b-bit binary number in [0, 1]. Therefore the stochastic number S can be calculated by

$$S = \frac{1}{N_s}\sum_{i=1}^{N} y_i = \frac{1}{PMM\cdot 2^{N_b}}\sum_{i=1}^{N_s} y_i, \quad (70)$$

where N_s is the stochastic sequence length.

For a stochastic circuit, S_i ($i = 1, 2, \ldots, N_s$) is defined as a random variable to indicate if an error occurs or not in a stochastic bit stream, i.e., if S_i is 1, an error occurs, so bit i flips. The error injection rate is also considered to be ε, i.e.,

$$P(S_i = 1) = \varepsilon. \quad (71)$$

$S_1, S_2, \dots, S_{N_s}$ are statistically independent, so

$$E[S_i] = \varepsilon; \tag{72}$$

$$Var[S_i] = (1-\varepsilon)\varepsilon. \tag{73}$$

A bit affected by a possible error is denoted by y'_i, thus

$$y'_i = S_i(1-x_i) + x_i(1-S_i). \tag{74}$$

The stochastic number then becomes

$$S' = \frac{1}{N_s}\sum_{i=1}^{N} y'_i = \frac{1}{N_s}\sum_{i=1}^{N_s}[S_i(1-y_i) + y_i(1-S_i)]. \tag{75}$$

Hence, the stochastic error $e_s^{(i)}$ is determined by

$$e_s^{(i)} = S' - S = \frac{1}{N_s}\sum_{i=1}^{N_s} S_i(1-2y_i). \tag{76}$$

By (70), (72) and (76), the mean of error $e_s^{(i)}$ is given by

$$E\left[e_s^{(i)}\right] = \frac{1}{N_s}\sum_{i=1}^{N_s}(1-2y_i)\,E[S_i] = \frac{1}{PMM\cdot 2^{N_b}}\sum_{i=1}^{N_s}(1-2y_i)\,\varepsilon = (1-2S)\varepsilon. \tag{77}$$

With (73) and (76), the variance of error $e_s^{(i)}$ is obtained as

$$Var\left[e_s^{(i)}\right] = \frac{1}{N_s^2}\sum_{i=1}^{N_s}(1-2y_i)^2\,Var[S_i] = \frac{1}{N_s^2}\sum_{i=1}^{N_s}(1-2y_i)^2\,(1-\varepsilon)\varepsilon = \frac{1}{PMM\cdot 2^{N_b}}(1-\varepsilon)\varepsilon. \tag{78}$$

To investigate how injected errors affect the function to implement FIR filters, we assume that the injected errors $e_c^{(i)}$ and $e_s^{(i)}$ follow additive Gaussian distributions.

For the conventional binary FIR filter defined in (25) without injected errors, the erroneous output $Y^{(i)}$ is given by

$$Y^{(i)} = \sum_{i=0}^{N_f-1}(H_i + e_{Hi}^{(i)})(X_i + e_{Xi}^{(i)}), \tag{79}$$

where $e_{Hi}^{(i)}$ and $e_{Xi}^{(i)}$ are independent errors modeled by $e_{oc}^{(i)}$ with the mean and variance given in (68) and (69), respectively. The overall error due to error injection for the conventional binary FIR filters $e_{oc}^{(i)}$ is then

$$e_{oc}^{(i)} = Y^{(i)} - Y. \tag{80}$$

When the error injection rate ε is small, the mean and variance of the overall error $e_{oc}^{(i)}$ are given by (approximately)

$$E\left[e_{oc}^{(i)}\right] \approx \sum_{i=0}^{N_f-1}[(H_i + X_i - 4H_iX_i)\varepsilon + (1-2H_i)(1-2X_i)\varepsilon^2]; \tag{81}$$

$$Var\left[e_{oc}^{(i)}\right] \approx \frac{\varepsilon}{9}\sum_{i=0}^{N_f-1}\{3(H_i^2 + X_i^2) + [1-3(H_i^2+X_i^2)]\varepsilon\}. \tag{82}$$

How (81) and (82) are derived is shown in detail in the appendix. Similarly, the overall error due to error injection for the stochastic FIR filters $e_{os}^{(i)}$ is given by

$$e_{os}^{(i)} = P_{Y^{(i)}} - P_Y. \tag{83}$$

Its mean and variance are given by

$$E\left[e_{os}^{(i)}\right] = \sum_{i=0}^{N_f-1}[(P_{H_i} + P_{X_i} - 4P_{H_i}P_{X_i})\varepsilon + (1-2P_{H_i})(1-2P_{X_i})\varepsilon^2]; \tag{84}$$

$$Var\left[e_{os}^{(i)}\right] \approx \frac{\varepsilon}{N_s^2}\sum_{i=0}^{N_f-1}\{N_s(H_i^2 + X_i^2) + [1-N_s(P_{Xi}^2 + P_{Hi}^2)]\varepsilon\}. \tag{85}$$

$E\left[e_{oc}^{(i)}\right]$ and $E\left[e_{os}^{(i)}\right]$ in (81) and (84) show that the mean output error depends on both the inputs and the coefficients. The mean of the stochastic error $E\left[e_{os}^{(i)}\right]$ is identical to the mean of the conventional binary error $E\left[e_{oc}^{(i)}\right]$ for the same filter function (thus with the same inputs and coefficients).

To compare the variances of the binary error and the stochastic error, all the inputs and coefficients in (82) and (85) are assumed to be 0.5. We then obtain

$$Var\left[e_{oc}^{(i)}\right] \approx \frac{5N_f}{18}\varepsilon - \frac{N_f}{6}\varepsilon^2, \tag{86}$$

$$Var\left[e_{os}^{(i)}\right] \approx \left(\frac{N_f}{N_s^2} + \frac{N_f}{2N_s}\right)\varepsilon - \frac{N_f}{2N_s}\varepsilon^2. \tag{87}$$

For any $N_s > 3$, the variances in (86) and (87) satisfy

$$Var\left[e_{oc}^{(i)}\right] > Var\left[e_{os}^{(i)}\right]. \tag{88}$$

Due to the factor of N_s, the stochastic method results in a smaller variance. The variation of the error for the stochastic implementation $e_{os}^{(i)}$ is inversely proportional to the sequence length squared, N_s^2. When using $N_s = PMM \cdot 2^{N_b}$ ($PMM = 2^0, 2^1, 2^2, \dots$) bits in the stochastic encoding of an N_b-bit binary number, the variance of the stochastic error can be reduced by increasing the PMM. In the conventional binary approach, however, it is more difficult to obtain a smaller variance as it lacks the tuning parameter PMM in the stochastic approach.

B. Fault-tolerance Simulation

Simulations are further performed to evaluate the reliability of the binary and stochastic circuits. To measure the reliability of a design, the average absolute error (AAE) is defined as

$$AAE = \frac{1}{M}\cdot\frac{1}{2^{2N_b+4}}\sum_{i=0}^{M-1}|X_i - X'_i|, \tag{89}$$

where X_i and X'_i are the expected correct output and the actual output, respectively, M is the number of simulations, and the factor $\frac{1}{2^{2N_b+4}}$ is taken as a constant coefficient so that the AAEs are between 0 and 1 ($N_b = 13$ here). The AAE indicates how seriously the injected error affects the correct output.

In the fault tolerance analysis, the effect of faults is also taken into account in auxiliary circuits such as the SNGs and counters. Digital logic is modeled in Matlab such that bit flips can occur at either the input or output. A bit flips with a certain probability as indicated by the error injection rate. The consequence of the bit flips can be seen at the final output. How the faults affect the filter behavior is reflected by the value of AAE. We investigate the AAE for the conventional binary 13-bit low-pass FIR filter with 267 taps, as well as the stochastic MWA design and HWA design using a sequence length of 4,194,304 bits (from Table 3) under various injected error rates. In addition, redundant copies of the binary circuit can be used to achieve a better fault tolerance, for example, in the form of triple modular redundancy (TMR). We further consider the TMR implementations of the binary circuit with unreliable and fault-free voters [18]. The stochastic computational models in [12, 13] are used to facilitate our fault-tolerance analysis. XOR gates are used to inject errors into the circuit. The majority voters in the TMR circuits are considered bitwise rather than word-wise.

Table 8 shows the comparison of AAEs obtained from 200 simulations with a sequence length of 100,000 bits. The results with error injection are compared with those without error injection, thus the AAEs for the stochastic circuits are 0 when the injected error rate is 0. It can be seen that the AAE increases as the injected error rate increases. The conventional binary circuit is not as fault-tolerant as the stochastic circuits, which is consistent with the analysis. When one bit in a binary circuit flips, it can cause a serious error if the erroneous bit is among the MSBs. However, all bits in a stochastic sequence have the

same weight, so the effect of a single bit flip is insignificant in a relatively long stochastic sequence. The binary TMR circuit with unreliable voters has an improved reliability, but it is still not as reliable as the stochastic approaches. However, the binary TMR circuit with reliable voters becomes more fault-tolerant than the stochastic circuits.

VIII. Conclusions

In this paper, a stochastic hardwired weighted average (HWA) design and a multi-level weighted average (MWA) design are proposed for implementing FIR filters. The HWA design takes advantage of simply repeating the input wires of a multiplexer to implement the weights of different data inputs, while the MWA design uses multiplexers to generate the required filter coefficients or weights. The proposed stochastic designs show an improved performance, a smaller circuit area and lower power consumption, compared with the conventional stochastic design. The MWA design with multiple stages is not as competitive as the HWA design, but it can be more easily reconfigured by re-programming the weight generators. This task is not easy for the HWA design that uses repeated inputs to implement the filter coefficients.

Table 8. Average absolute error of the stochastic and binary circuits with and without redundancy at various injected error rates. The results are obtained from 200 simulations using sequences of 100,000 bits.

Error Rate (%)	Average Absolute Error (%)				
	MWA	HWA	Binary	Binary TMR	
				Error-free Voter	Unreliable Voter
0	0	0	0	0	0
0.1	0.063	0.065	1.507	0.004	0.126
0.2	0.121	0.136	2.325	0.009	0.225
0.5	0.339	0.326	3.290	0.035	0.581
1	0.592	0.574	5.209	0.111	1.203
2	1.476	1.226	6.368	0.198	2.382
5	3.009	2.948	10.942	0.337	5.794
10	5.472	5.696	21.477	1.123	12.050

Compared to binary FIR filter circuits, the proposed stochastic designs have a significant advantage in circuit area, especially at higher resolutions. With respect to the performance metrics of throughput per area and energy per operation, however, the stochastic design does not show any advantages over its binary counterpart. This is primarily due to the significant latency in stochastic computing because long stochastic sequences must be used to achieve the same filtering performance as a binary circuit. With a shorter stochastic sequence, however, the stochastic circuit shows a graceful degradation in performance compared to the binary design. The features of a stochastic circuit are investigated in detail by both analysis and simulation.

A binary TMR circuit using error-free voters is shown to be more reliable than the stochastic design. Due to its intrinsic fault tolerance, however, the proposed stochastic design shows significant advantages in reliability over the conventional binary design and its TMR implementation when the voters are subject to errors. These results suggested that other sum-of-product based circuits could also benefit from stochastic implementation.

Acknowledgement

This work was partly supported by the University of Alberta and the Natural Sciences and Engineering Research Council (NSERC) of Canada.

Appendix

In this appendix, we prove that (81) and (82) give the mean and variance of the overall error in the conventional binary filter circuit. To investigate how the injected errors affect the output of the FIR filters, we assume that the correct output of the conventional binary FIR filter Y is given by (25), where H_i and X_i are the inputs and filter coefficients without injected error. With error injection, the output $Y^{(i)}$ given in (79) is evaluated by

$$\begin{aligned} Y^{(i)} &= \sum_{i=0}^{N_f-1}(H_i + e_{Hi}^{(i)})(X_i + e_{Xi}^{(i)}) \\ &= \sum_{i=0}^{N_f-1}(H_iX_i + X_ie_{Hi}^{(i)} + H_ie_{Xi}^{(i)} + e_{Hi}^{(i)}e_{Xi}^{(i)}) \\ &= Y + \sum_{i=0}^{N_f-1}(X_ie_{Hi}^{(i)} + H_ie_{Xi}^{(i)} + e_{Hi}^{(i)}e_{Xi}^{(i)}), \end{aligned} \tag{90}$$

where $e_{Hi}^{(i)}$ and $e_{Xi}^{(i)}$ are statistically independent errors of Gaussian distribution for the coefficient H_i and the input X_i with mean and variance given by (68) and (69), respectively. Then we have

$$E\left(e_{Hi}^{(i)}\right) \approx (1-2H_i)\varepsilon; \tag{91}$$

$$E\left(e_{Xi}^{(i)}\right) \approx (1-2X_i)\varepsilon; \tag{92}$$

$$Var\left[e_{Xi}^{(i)}\right] = Var[e_{Hi}^{(i)}] \approx \tfrac{1}{3}(1-\varepsilon)\varepsilon. \tag{93}$$

The overall error due to error injection for the conventional binary FIR filter $e_{oc}^{(i)}$ is given by (80). The mean of the overall error $e_{oc}^{(i)}$ is given by

$$\begin{aligned} \mathrm{E}\left(e_{oc}^{(i)}\right) = E(Y'-Y) = E\Big(\sum_{i=0}^{N_f-1}(X_ie_{Hi}^{(i)} + H_ie_{Xi}^{(i)} + \\ e_{Hi}^{(i)}e_{Xi}^{(i)})\Big) = \sum_{i=0}^{N_f-1}[X_iE\left(e_{Hi}^{(i)}\right) + H_iE\left(e_{Xi}^{(i)}\right) + \\ E\left(e_{Hi}^{(i)}e_{Xi}^{(i)}\right)]. \end{aligned} \tag{94}$$

Since $e_{Hi}^{(i)}$ and $e_{Xi}^{(i)}$ are statistically independent, the mean of their product is [15]

$$E\left(e_{Hi}^{(i)}e_{Xi}^{(i)}\right) = E\left(e_{Hi}^{(i)}\right)E\left(e_{Xi}^{(i)}\right) \approx (1-2H_i)(1-2X_i)\varepsilon^2. \tag{95}$$

By (91), (92) and (95), it is easy to show that the mean of the overall error of the conventional binary circuit due to error injection is given by (81).

The variance of the overall error $e_{oc}^{(i)}$ is given by

$$\begin{aligned} Var\left[e_{oc}^{(i)}\right] = Var\Big(\sum_{i=0}^{N_f-1}(X_ie_{Hi}^{(i)} + H_ie_{Xi}^{(i)} + e_{Hi}^{(i)}e_{Xi}^{(i)})\Big) = \\ \sum_{i=0}^{N_f-1}[X_i^2Var(e_{Hi}^{(i)}) + H_i^2Var(e_{Xi}^{(i)}) + Var(e_{Hi}^{(i)}e_{Xi}^{(i)})]. \end{aligned} \tag{96}$$

The variance of the product of the two independent variables $e_{Hi}^{(i)}$ and $e_{Xi}^{(i)}$ can be calculated by

$$\begin{aligned} Var\left(e_{Hi}^{(i)}e_{Xi}^{(i)}\right) = Var\left(e_{Hi}^{(i)}\right)Var\left(e_{Xi}^{(i)}\right) - Var\left(e_{Hi}^{(i)}\right)E^2\left(e_{Xi}^{(i)}\right) \\ - Var\left(e_{Xi}^{(i)}\right)E^2\left(e_{Hi}^{(i)}\right). \end{aligned} \tag{97}$$

By (91), (92) and (93), equation (97) can be further written as

$$\begin{aligned} Var\left(e_{Hi}^{(i)}e_{Xi}^{(i)}\right) = \tfrac{1}{9}\,\varepsilon^2(1-\varepsilon)^2 - \tfrac{1}{3}\,\varepsilon^3(1-\varepsilon)[(1- \\ 2H_i)^2 + (1-2X_i)^2]. \end{aligned} \tag{98}$$

Due to (93) and (98), equation (96) becomes

$$Var\left[e_{oc}^{(i)}\right] = \sum_{i=0}^{N_f-1}\left\{\frac{1}{3}(H_i^2 + X_i^2)\varepsilon(1-\varepsilon) + \frac{1}{9}\varepsilon^2(1-\varepsilon)^2 - \frac{1}{3}\varepsilon^3(1-\varepsilon)[(1-2H_i)^2 + (1-2X_i)^2]\right\}. \quad (99)$$

When the injected error rate ε is small, ε^k for $k \geq 3$ in (99) can be ignored. This immediately leads to the variance given in (82).

REFERENCES

[1] B. R. Gaines, "Stochastic computing systems." In Advances in Information Systems Science, pp. 72-73, Springer US, 1969.

[2] A. Alaghi and J.P. Hayes, "Survey of stochastic computing." *ACM Trans. on Embedded Computing Systems (TECS)* 12, no. 2s (2013): 92.

[3] P. Li and D. J. Lilja, "Using stochastic computing to implement digital image processing algorithms." *IEEE ICCD*, pp. 154-161, 2011.

[4] P. Li, D. J. Lilja, W. Qian, K. Bazargan, and M. Riedel, "Computation on stochastic bit streams digital image processing case studies," *IEEE Trans. on VLSI Systems*, vol. 22, no. 3, pp. 449–462, March 2014.

[5] A. Alaghi, C. Li, and J. P. Hayes, "Stochastic circuits for real-time image-processing applications." *DAC*, pp. 1-6, 2013.

[6] W. Qian and M. D. Riedel, "The synthesis of robust polynomial arithmetic with stochastic logic." *DAC 2008.* pp. 648-653, 2008.

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

[8] J. Chen and J. Hu, "A novel FIR filter based on stochastic logic," Circuits and Systems (ISCAS), 2013 IEEE International Symposium on , vol., no., pp.2050,2053, 19-23 May 2013.

[9] Saraf, Naman, Kia Bazargan, David J. Lilja, and Marc D. Riedel. "IIR filters using stochastic arithmetic." In *Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pp. 1-6. IEEE, 2014.

[10] R. Wang, J. Han, B. Cockburn, and D. Elliott, "Design and Evaluation of Stochastic FIR Filters," in *Proc. 2015 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, Victoria, BC, Canada, August 24 - 26, 2015.

[11] B. Moons and M. Verhelst, "Energy and Accuracy in Multi-Stage Stochastic Computing." *The 12th IEEE International New Circuits and Systems Conference*, Trois-Rivières**,** Canada, 22-25 June 2014.

[12] J. Han, H. Chen, J. Liang, P. Zhu, Z. Yang, and F. Lombardi, "A stochastic computational approach for accurate and efficient reliability evaluation." *IEEE Trans. on Computers*, vol. 63, no. 6, pp. 1336 – 1350, June 2014.

[13] H. Chen and J. Han, "Stochastic computational models for accurate reliability evaluation of logic circuits," in *GLSVLSI'10, Proceedings of the 20th IEEE/ACM Great Lakes Symposium on VLSI*, Providence, Rhode Island, USA, pp. 61–66, 2010.

[14] B. Widrow and István Kollár, "Roundoff Noise in FIR Digital Filters and in FFT Calculations." *Quantization Noise: Roundoff Error in Digital Computation, Signal Processing, Control, and Communications*, pp. 373-382, Cambridge, 2008

[15] Qian, Weikang, Xin Li, Marc D. Riedel, Kia Bazargan, and David J. Lilja. "An architecture for fault-tolerant computation with stochastic logic." *IEEE Transactions on Computers, vol.* 60, no. 1 (2011): 93-105.

[16] Ma, Chengguang, Shunan Zhong, and Hua Dang. "Understanding variance propagation in stochastic computing systems." In *2012 IEEE 30th International Conference on Computer Design (ICCD)*, pp. 213-218.

[17] Li, P., Lilja, D. J., Qian, W., Riedel, M. D., and Bazargan, K. (2014). Logical Computation on Stochastic Bit Streams with Linear Finite-State Machines. IEEE Transactions on Computers, 63(6), 1474-1486.

[18] T. Ban and L. Naviner. "Progressive module redundancy for fault-tolerant designs in nanoelectronics." *Microelectronics Reliability* 51, no. 9 (2011): 1489-1492.