electronics

MDPI

Article

A High-Accuracy Stochastic FIR Filter with Adaptive Scaling Algorithm and Antithetic Variables Method

Ying Zhang [1], Yubin Zhu [2], Kaining Han [2,*], Junchao Wang [1] and Jianhao Hu [2,*]

[1] Department of Electrical Engineering, Shantou University, Shantou 515063, China; y_zhang2021@126.com (Y.Z.); junchaowang@stu.edu.cn (J.W.)
[2] National Key Lab of Science and Technology on Communications, University of Electronic Science and Technology of China, Chengdu 611731, China; ybzhu2020@foxmail.com
* Correspondence: hkaining2014@gmail.com (K.H.); jhhu@uestc.edu.cn (J.H.)

Abstract: Digital filter is an important fundamental component in digital signal processing (DSP) systems. Among the digital filters, the finite impulse response (FIR) filter is one of the most commonly used schemes. As a low-complexity hardware implementation technique, stochastic computing has been applied to overcome the huge hardware cost problem of high-order FIR filters. However, the stochastic FIR filter (SFIR) scheme suffers from long processing latency and accuracy degradation. In this paper, the bit stream representation noise is theoretically analyzed, and an adaptive scaling algorithm (ASA) is proposed to improve the accuracy of SFIR with the same bit stream length. Furthermore, a novel antithetic variables method is proposed to further improve the accuracy. According to the simulation results on a 64-tap FIR filter, the ASA and AV methods gain 17 dB and 6 dB on the signal-to-noise ratio (SNR), respectively. The hardware implementation results are also presented in this paper, which illustrates that the proposed ASA-AV-SFIR filter increases 4.6 times hardware efficiency with respect to the existing SFIR schemes.

Keywords: stochastic computing; FIR filter; adaptive scaling algorithm; antithetic variables method

Citation: Zhang, Y.; Zhu, Y.; Han, K.; Wang, J.; Hu, J. A High-Accuracy Stochastic FIR Filter with Adaptive Scaling Algorithm and Antithetic Variables Method. *Electronics* **2021**, *10*, 1937. https://doi.org/10.3390/electronics10161937

Academic Editor: Leonardo Pantoli

Received: 5 July 2021
Accepted: 3 August 2021
Published: 11 August 2021

1. Introduction

Digital filter is an important fundamental component in digital signal processing (DSP) systems such as image processing [1], speech signal processing, and communication systems [2]. In particular, the finite impulse response (FIR) filter is one of the basic and most commonly used digital filters due to its linear phase feature. The major challenge of high-throughput FIR filter implementation is the huge hardware cost, especially for high-order filter applications.

Stochastic computing (SC) is a low-complexity hardware implementation technique, which has been widely used in communication systems [3], image processing systems [4], and support vector machines [5]. In the existing schemes, stochastic computing has been applied to the FIR filter to reduce the hardware cost and critical path delay. Different from the conventional 2's complement system (TCS), the input signal and coefficients are represented with stochastic bit streams in SC-based FIR filters. As a result, the complex arithmetic operations in TCS circuits can be mapped into quite simple logic gates operation [2,6–11]. Reference [6] proposes a bipolar mapping scheme and presents a complete stochastic FIR filter architecture, where the XNOR gate can implement the multiplication. To further reduce the hardware cost, a pseudo-random number sharing method is proposed in [12], reducing the total number of random number generators in the SFIR. The SFIR filter shows advantages in the extremely low hardware cost. While it still suffers from the long processing latency and accuracy degradation due to the relatively long stochastic bit streams [4]. To improve the calculation accuracy of SFIR filters, Reference [2] proposed a two-line mapping scheme, where the sign and magnitude are represented with two bit streams, respectively, and demonstrates obvious accuracy gains. In [7], a hybrid scheme is

proposed, where the multiplication is implemented with stochastic logic and the addition is still in the TCS manner. In [13,14], it was observed that stochastic computing suffers from low accuracy for small number values. Thus, a scaling method was utilized on the filter coefficient, which scales up the coefficients to achieve higher accuracy. However, the scaling method in [14] is quasi-static and coefficient-only processing, which cannot applied on the real-time input signals directly. Furthermore, there is still a lack of theoretical analysis.

In this paper, a high-accuracy stochastic FIR filter with adaptive scaling algorithm and antithetic variable method is proposed. The main contributions are as follows:

- The relationship between representation noise and represented values of a stochastic bit stream is theoretically analyzed, and it is found that stochastic computing can achieve high accuracy in certain value intervals, providing a potential way to improve the accuracy even with the same stochastic bit stream length.
- An adaptive scaling algorithm (ASA) is proposed for the SFIR to scale both the input signals and coefficients into low-noise regions.
- A novel antithetic variables method (AV) is proposed to further improve the accuracy, and the theoretical proof is also provided.
- The hardware architecture of the proposed ASA–SFIR and ASA–AV–SFIR is designed and implemented, which demonstrates high-accuracy performance advantages with respect to the existing SFIR filters.

The remainder of this paper is organized as follows. Section 2 introduces the theoretical background of FIR filter, stochastic computing, and stochastic filter designs. Afterward, the proposed design with ASA and AV is presented in Section 3. Section 4 demonstrates the performance evaluation and hardware implementation of the proposed design. The last section concludes the performed work and discusses the potential future work.

2. Theoretical Background

In this section, the background of FIR filter, stochastic computing, and the existing SFIR filters are introduced.

2.1. FIR Filter

The FIR filter is one of the most commonly used digital filters in digital signal processing systems. In general, the output signal $y[n]$ of a K-tap FIR filter can be calculated in the time domain as (1), where $x[n]$ is the input discrete-time signal, c_i is the filter coefficient, and K is the tap of the FIR filter.

$$y[n] = \sum_{i=1}^{K} x[n-i] \cdot c_i \tag{1}$$

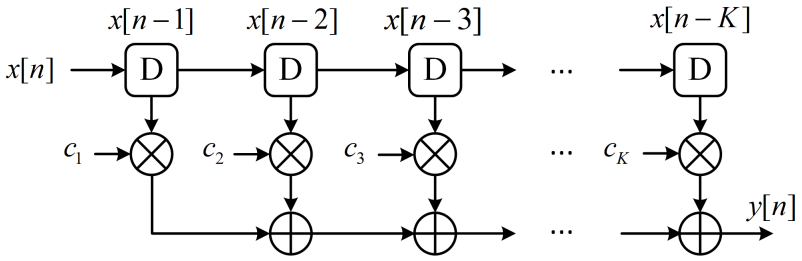

Figure 1. Conventional hardware architecture of FIR filter [2].

In the practical hardware implementation, the filter coefficient c_i is usually normalized as (2) to keep the dynamic range of output signals the same with input signals and avoid the calculation overflow error. The filter coefficients are all normalized in the following.

$$c_i^{\text{norm}} = c_i / \sum_{j=1}^{K} |c_j| \tag{2}$$

It can be observed that K multipliers and $K-1$ adders are required for a K-tap FIR filter in the conventional hardware implementation schemes, illustrated as Figure 1. It would be a huge complexity for practical DSP systems with high-order FIR filters.

2.2. Stochastic Computing

Stochastic computing is a low complexity algorithm design and hardware implementation technique, where a numerical value a is represented with stochastic bit stream A_i, $i = 1, 2, ..., N$. As a result, the complex operations in conventional TCS systems can be mapped into quite simple logic operations. A typical stochastic computing-based DSP system is illustrated in Figure 2.

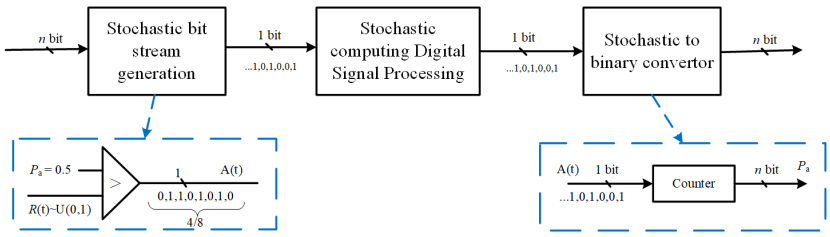

Figure 2. Typical-Stochastic-Computing-based DSP system [15].

- For a numerical value $a \in [0, 1]$, the unipolar format can be utilized to transform it to bit stream A_i by comparing it with a uniform distributed random number $R(t) \sim U(0, 1)$, where $\Pr\{A_i = 1\} = a$. The corresponding hardware architecture of unipolar format stochastic computing is shown as "stochastic bit stream generation" in Figure 2.
- For a numerical value $a \in [-1, 1]$, the bipolar format bit stream generation is required [4,11], where $\Pr\{A_i = 1\} = (a + 1)/2$. The bipolar format bit stream generation could share a similar hardware architecture with unipolar format by comparing $(a + 1)/2$ with the random number $R(t)$.

The stochastic computing-based logic circuits could significantly reduce the hardware cost and critical path delay compared with conventional TCS circuits, illustrated as Figure 3a, where the multiplication $P_c = P_a \cdot P_b$ is implemented by a single "AND" logic [16]. As shown in Figure 3b, a simple "XOR" logic is used to perform $P_c = P_a \cdot (1 - P_b + (1 - P_a) \cdot P_b)$ calculation. The scaled addition $P_c = P_a \cdot P_s + P_b \cdot (1 - P_s)$ can be realized by a multiplexer logic shown as Figure 3c, and the sum is scaled according to P_s. Except for the linear computation, addition and multiplication, the division $P_c = P_a/(P_a + P_b)$ can be implemented by a J-K Flip-Flop illustrated as Figure 3d. The backward-transformed numerical value is shown as "Stochastic to binary convertor" in Figure 2.

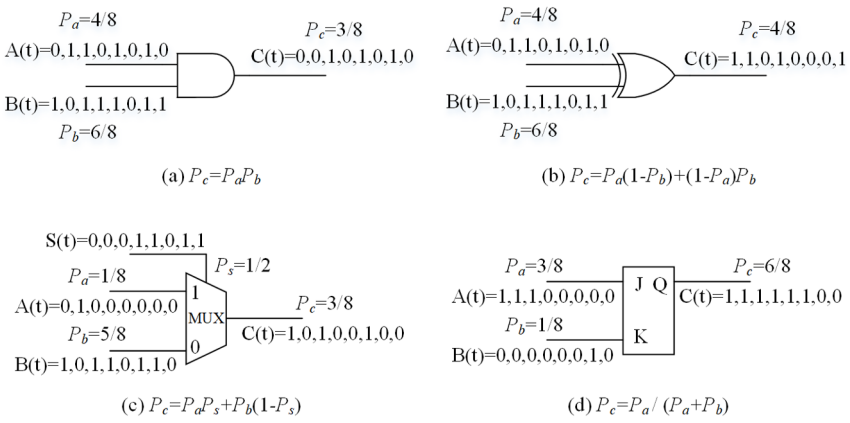

Figure 3. Basic calculation units of unipolar format stochastic computing [17].

2.3. Stochastic FIR Filter

To reduce the hardware cost of the FIR filter, a bipolar format SFIR filter is proposed in [6], where input signals and coefficients are normalized into the range [−1,1] and transformed into bit streams. As a result, the multiplication and addition involved in the FIR filter are mapped into "XNOR" logic and multiplexer, respectively. The bipolar-based SFIR filter has an extremely low hardware costs. However, the main drawback is the degradation of accuracy.

To overcome the accuracy degradation problem, a two-line scheme-based SFIR filter is proposed in [2], where each numerical value is represented with two stochastic bit streams: one is sign bit-stream and the other is magnitude bit-stream. The multiplication of two magnitude bit-streams and sign bit-streams are mapped into "AND" logic and "XOR" logic, respectively. The addition is implemented with a novel non-scaled two-line adder. The two-line scheme outperforms the bipolar scheme on accuracy with comparable hardware cost. However, there is still a relatively large gap with the ideal performance.

3. Stochastic FIR Filter with Adaptive Scaling

In this section, the representation noise of a stochastic bit stream is firstly analyzed in Section 3.1. Afterward, an adaptive scaling algorithm (ASA) is proposed in Section 3.2. Furthermore, the antithetic variables (AV) method is introduced in Section 3.3. Finally, ASA–AV–SFIR filter architecture is presented in Section 3.4.

3.1. Noise Analysis of Stochastic Bit Stream

Consider a numerical value $P \in [0,1]$ represented with a stochastic bit stream X_i, $i = 1, 2, ..., N$. Each bit in the stochastic bit stream follows Bernoulli distribution, and the variance of each bit can be written as $D(X_i) = E[(X_i - P)^2] = P \cdot (1 - P)$. When transformed back to a binary system, the estimated numerical value $\hat{P}$ and the corresponding variance can be written as (3) and (4), respectively.

$$\hat{P} = \frac{1}{N} \cdot \sum_{i=1}^{N} X_i \tag{3}$$

$$D(\hat{P}) = \frac{N \cdot D(X_i)}{N^2} = \frac{P \cdot (1 - P)}{N} \tag{4}$$

For the stochastic bit stream representation, the representation noise power P_{noise} can be calculated as (5). It can be observed that the noise power P_{noise} decreases with the increasing bit stream length N. In addition, the noise power P_{noise} also relies on the numerical value P. A bit-by-bit stochastic FIR filter simulation is operated using Matlab software, and the theoretical and simulation results are shown as Figure 4a, which demonstrates that the noise power is much lower when the numerical value P approaches 0 or 1.

$$P_{noise} = E[(P - \hat{P})^2] = D(\hat{P}) = \frac{P \cdot (1 - P)}{N} \tag{5}$$

Furthermore, the signal-noise ratio (SNR) can be calculated as (6), where the signal power $P_s = P^2$. It can be observed that the SNR would be affected by the numerical value P. Thus, it would be possible to increase the SNR by scaling the numerical value. In other words, when P is scaled up, the SNR would be increased even with the same bit stream length.

$$\text{SNR} = 10 \cdot \log_{10} \frac{P_s}{P_{noise}} = 10 \cdot \log_{10} \frac{N \cdot P}{1 - P} \tag{6}$$

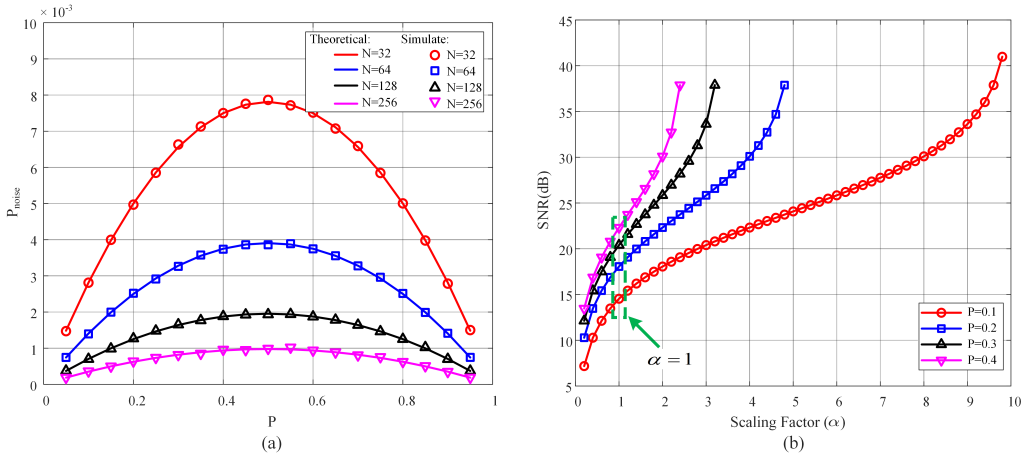

Figure 4. (**a**) MSE performance of stochastic bit stream representation with different bit stream length. (**b**) SNR performance of the stochastic bit stream representation for $P = 0.1, 0.2, 0.3, 0.4$ with scaling as (8).

3.2. Adaptive Scaling Algorithm

To improve the calculation accuracy of SFIR filters, a scaling method has been utilized on the filter coefficients [13,14], which demonstrates improvements in accuracy. However, the scaling is quasi-static coefficient-only processing and cannot be applied to the real-time input signals directly. Based on the noise analysis in Section 3.1, a new adaptive scaling algorithm is proposed in this section, where both the input signal and the filter coefficients are adaptively scaled to the high accuracy region of stochastic computing.

As shown in (6), SNR improves with the increasing numerical value P. Thus, a scaling factor α is set to scale up the numerical value $P' = \alpha \cdot P$ before stochastic bit stream generation. The SNR can be re-written as (7). To reduce the hardware complexity of the scaling operation, α is selected in $\alpha = 2^{\beta}$ manner, where β is shown as (8). As a result, the scaling operation can be implemented with shift registers. After scaling operation, the scaled numerical value P' can be transformed into a stochastic bit stream.

$$\mathrm{SNR} = 10 \cdot \log_{10} \frac{N \cdot P'}{1 - P'} = 10 \cdot \log_{10} \frac{N \cdot \alpha P}{1 - \alpha P} \tag{7}$$

$$P' = 2^{\beta} \cdot P \leq 1 \Rightarrow \beta = \left\lfloor \log_2 \frac{1}{P} \right\rfloor \tag{8}$$

After the bit-stream generation and stochastic computing-based operation, the output numerical value has to be re-scaled, which can also be implemented with shift registers.

Combing the scaling and re-scaling operation, the adaptive scaling algorithm (ASA) is applied to an "AND" logic to implement multiplication $z = x \cdot c$, shown as Algorithm 1. Note that the "Find scaling factor" step can also be implemented by shift registers, which would be presented in the next section.

Algorithm 1 Adaptive Scaling Algorithm (ASA)

Input: $x, c \in [0, 1]$
Output: $z = x \cdot c$
1: Find scaling factor: $\beta_x = \lfloor \log_2 1/x \rfloor, \beta_c = \lfloor \log_2 1/c \rfloor$
2: Scaling: $x << \beta_x, c << \beta_c$
3: Bit Stream Generation: $x, c \Rightarrow X_i, C_i, i = 1, 2, ..., N$
4: AND Logic: $Z'_i = X_i \cap C_i$
5: Bit-wise Re-scaling: $Z_i = Z'_i >> (\beta_x + \beta_c)$
6: Transform Back: $\hat{z}' = \sum_{i=1}^{N} Z_i / N$

3.3. Antithetic Variables Method

The parallelism method is a widely used method for stochastic computing to make a trade-off between processing latency and hardware cost. Consider a numerical value $P \in [0,1]$ represented with a stochastic bit stream X_i, $i = 1, 2, ..., N$. It takes N clock cycles to process the N bits in the bit stream. To reduce the processing latency, the bit stream X_i can be separate into two parts: X_i^1 and X_i^2, $i = 1, 2, ..., N/2$, where $X_i^1 = X_i$ and $X_i^2 = X_{i+N/2}$. The estimated $\hat{P}$ can be written as

$$\hat{P} = \frac{1}{N} \cdot \sum_{i=1}^{N} \frac{X_i^1 + X_i^2}{2} \tag{9}$$

For the reason that each bit in the stochastic bit stream follows Bernoulli distribution and is individual to the others, X_i^1 is individual to X_i^2 and $Cov(X_i^1, X_i^2) = 0$. Thus, the variance of $(X_i^1 + X_i^2)/2$ can be written as

$$\begin{aligned} D(\frac{X_i^1 + X_i^2}{2}) &= \frac{D(X_i^1) + D(X_i^2) + 2Cov(X_i^1, X_i^2)}{4} \\ &= \frac{D(X_i) + Cov(X_i^1, X_i^2)}{2} = \frac{D(X_i)}{2} \end{aligned} \tag{10}$$

It can be observed from (10) that the variance is reduced by 2 times. Combining (10) and (4), it demonstrates that the calculation accuracy can be improved by 2 times using the 2-parallelism method. However, the hardware cost is also 2 times higher, which is required by the parallel processing.

In this paper, an novel antithetic variables method is proposed to further improve the calculation accuracy. The basic idea is generate a certain bit stream X_i^2 to make $Cov(X_i^1, X_i^2) < 0$:

$$\begin{aligned} X_i^1 &= \begin{cases} 1, & P \geq R(t); \\ 0, & otherswise; \end{cases} \\ X_i^2 &= \begin{cases} 1, & P \geq 1 - R(t); \\ 0, & otherswise; \end{cases} \end{aligned} \tag{11}$$

where $R(t) \sim U(0,1)$ and $1 - R(t) \sim U(0,1)$. The expectation of X_i^1 and X_i^2 can be written as,

$$\begin{aligned} E(X_i^1) &= \int_0^P 1 dr \\ E(X_i^2) &= \int_{1-P}^1 1 dr \end{aligned} \tag{12}$$

Based on (12), the covariance $Cov(X_i^1, X_i^2)$ can be written as (13).

$$\begin{aligned} Cov(X_i^1, X_i^2) &= E(X_i^1 X_i^2) + E(X_i^1)E(X_i^2) \\ &= \begin{cases} 0 + E(X_i^1)E(X_i^2), & P < 0.5; \\ \int_{1-P}^P 1 dr + E(X_i^1)E(X_i^2), & P \geq 0.5; \end{cases} \\ &= \begin{cases} -P^2, & P < 0.5; \\ -P^2 + 2P - 1, & P \geq 0.5; \end{cases} \end{aligned} \tag{13}$$

Similar with the analysis in Section 3.1, X_i^1 and X_i^2 both follow Bernoulli distribution, and the variance of X_i^1 and X_i^2 can be directly presented as (14). Combining (13) and (14), the variance of $\frac{X_i^1 + X_i^2}{2}$ can be written as (15).

$$\begin{aligned} D(X_i^1) &= P(1 - P) \\ D(X_i^2) &= P(1 - P) \end{aligned} \tag{14}$$

$$D(\frac{X_i^1 + X_i^2}{2}) = \frac{D(X_i^1) + D(X_i^2) + 2Cov(X_i^1, X_i^2)}{4} = \begin{cases} (P - 2P^2)/2, & P < 0.5; \\ (-2P^2 + 3P - 1)/2, & P \geq 0.5; \end{cases} \tag{15}$$

Finally, the noise power of antithetic variables-method-based bit stream representation can be written as (16), and the corresponding simulation results are shown as Figure 5, which indicates that the simulation results agree with the theoretical analysis as (16).

$$\begin{aligned} P_{noise-AV} &= D\left(\frac{1}{N} \cdot \sum_{i=1}^{N} \frac{X_i^1 + X_i^2}{2}\right) \\ &= \frac{1}{N^2} \cdot \sum_{i=1}^{N} D(\frac{X_i^1 + X_i^2}{2}) \\ &= \begin{cases} (P - 2P^2)/(2 \cdot N), & P < 0.5; \\ (-2P^2 + 3P - 1)/(2 \cdot N), & P \geq 0.5; \end{cases} \end{aligned} \tag{16}$$

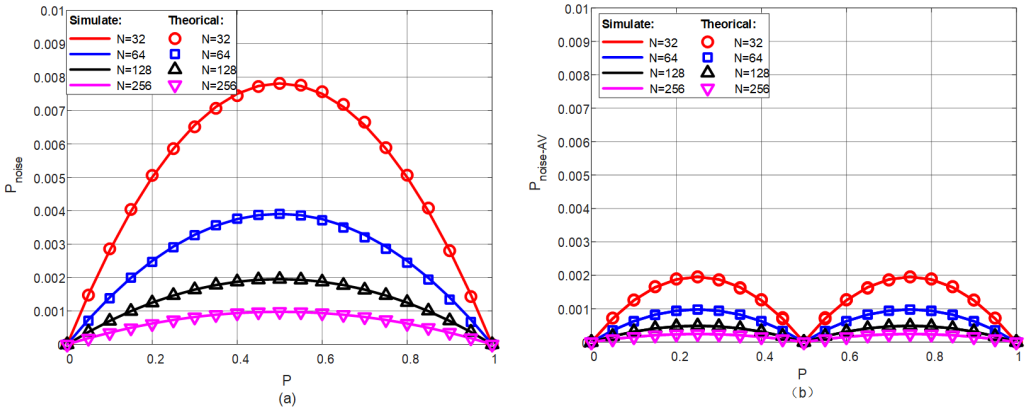

Figure 5. (**a**) MSE performance of stochastic bit stream representation (X_i, $i = 1, 2, ..., N$). (**b**) MSE performance of stochastic bit stream representation with AV (X_i^1 and X_i^2, $i = 1, 2, ..., N$).

3.4. Stochastic FIR Filter with ASA and AV

Applying the proposed ASA to the SFIR filter would be helpful to improve calculation accuracy and SNR of the output signal. The hardware architecture of the scaling module (SM) is illustrated in Figure 6, which corresponds to "Find scaling factor" and "Scaling" step in Algorithm 1. The scaling factor $\beta_x = \lfloor \log_2 1/x \rfloor$ is easy to find using the left-shift registers. As a general example, consider a input value $x = 0.21875$, whose sign bit $S(x) =$ "0" is extracted firstly. The magnitude value in binary format $|x| =$ "b'00111000" and initial scaling factor value in binary format $\beta_x =$ "b'00000001" are loaded in the left-shift registers. Afterward, all of the registers begin to left-shift until the first "1" occurs at the most significant bit (MSB), and the rest cycles are in an idle state. Note that the total number of left-shift cycles equals the date width of $|x|$. Finally, the scaled value $|x| \cdot 2^{\beta} = 0.875 =$ "b'11100000" and scaling factor $\beta_x = 2$ are output.

After scaling and serials of stochastic-logic-based operations, re-scaling module is required to realize the re-scaling operation, which is corresponding to the "Bit-wise Re-scaling" step in Algorithm 1. The architecture of re-scaling module (RSM) is shown as Figure 7, where the bit stream $X(t)$ is accumulated by a counter as,

$$cnt(t) = X(t) + cnt(t - 1). \tag{17}$$

Afterward, the counter $cnt(t)$ is compared with scaling factor β_x with "XNOR" logic. The output re-scaled bit stream can be represented as (18).

$$X'(t) = \begin{cases} 0, & cnt(t) \neq \beta_x; \\ 1, & cnt(t) = \beta_x; \end{cases} \tag{18}$$

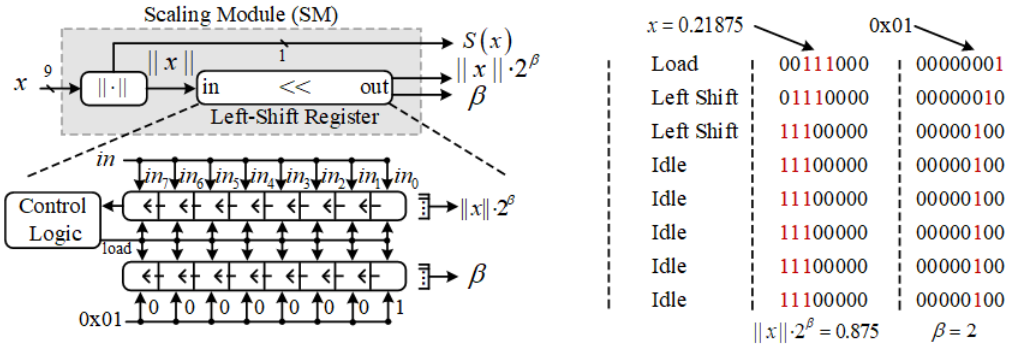

Figure 6. Hardware architecture of scaling modules.

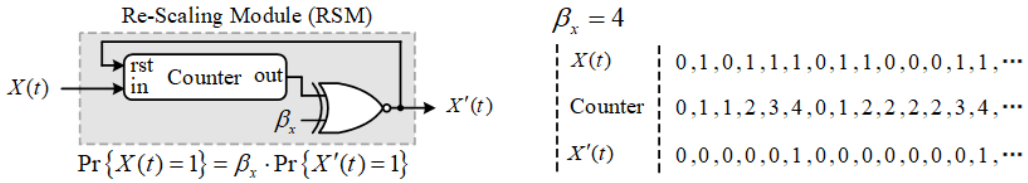

Figure 7. Hardware architecture of re-scaling modules.

As introduced in Section 1, the two-line scheme-based SFIR [2] outperforms the bipolar format-based scheme [6] on accuracy performance. Using the proposed ASA method, the accuracy performance of the two-line scheme-based SFIR can be further improved. The hardware architecture is shown as Figure 8, where ASA is applied in the scaling stochastic multiplication (SSM) module, involving scaling module (SM) and re-scaling module (RSM).

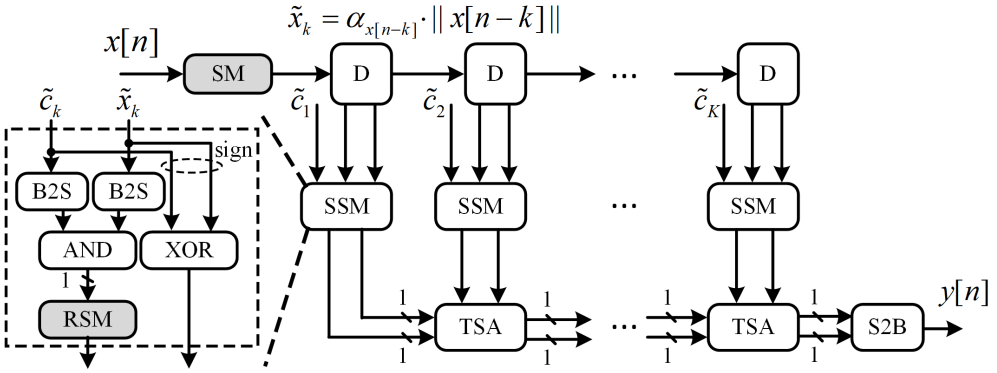

Figure 8. Hardware architecture of the proposed SFIR filter with scaling.

Except for the scaling module and re-scaling module, binary-to-stochastic converter (B2S) and stochastic-to-binary converter (S2B) are required to realize the conversion between binary numbers and stochastic bit streams. The two-line stochastic addition (TSA) module is similar to [2], which is a calculation-error-free addition scheme. The specific steps of the ASA-based SFIR filter are as follows:

Step 1: The FIR filter coefficient $c_k (k = 1, 2, ..., K)$ is initially scaled up to $\widetilde{c}_k$, while the input signal x_k is scaled up to $\widetilde{x}_k$ in real-time using SM module.

Step 2: In the scaled stochastic multiplication module (SSM), the sign bit $S(x_k)$ and $S(c_k)$ are extracted from $\widetilde{x}_k$ and $\widetilde{c}_k$, respectively, while the magnitude bit-streams $M(x_k)$, $M(c_k)$ are transformed from $\widetilde{c}_k$ and $\widetilde{x}_k$, respectively, using B2S module.

Step 3: Afterward, The multiplication on sign bit and magnitude bit-streams are mapped into "XOR" logic and "AND" logic, respectively. The bit-wise re-scaling operation is implemented by the RSM module.

Step 4: Finally, The outputs of the SSM module are summed up with the TSM module and transformed back to binary format using the S2B module.

Combining the proposed ASA and AV methods can further improve SNR of the output signal. The only difference between ASA based SFIR filter and ASA-AV based SFIR filter is the scaled stochastic multiplication module (SSM). The SSM of SFIR Filter with ASA is shown as Figure 9a and the SSM of SFIR Filter with ASA and AV is shown as Figure 9b.

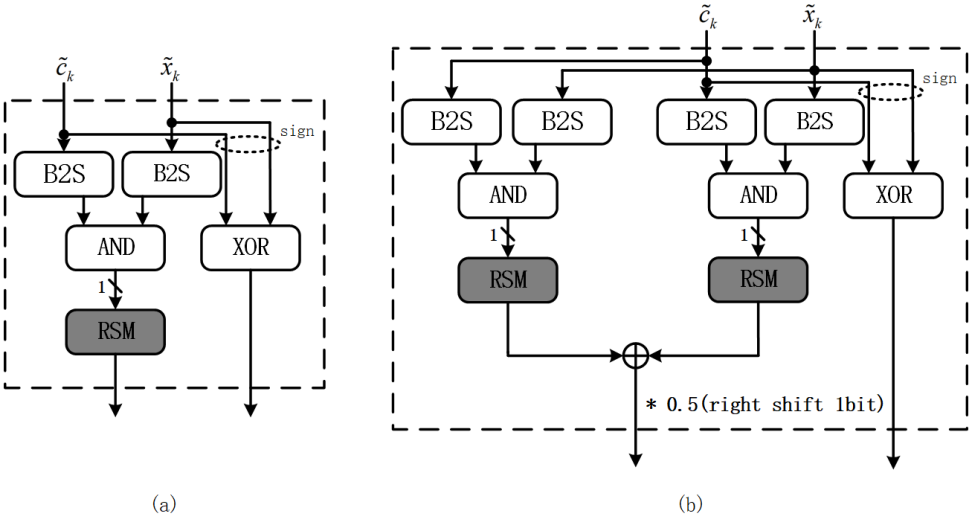

Figure 9. (**a**) The SSM of SFIR Filter with ASA. (**b**) The SSM of SFIR Filter with ASA and AV.

In the SSM of SFIR filter with ASA and AV, $\widetilde{c}_k$ can be transformed into magnitude bit-stream M1(c_k) and M2(c_k) by comparing it with rand numbers R1(t) and (1-R1(t)), respectively, and $\widetilde{x}_k$ can be transformed into magnitude bit-stream M1(x_k) and M2(x_k) by comparing it with rand numbers R2(t) and (1-R2(t)), respectively. Then the multiplication of M1(c_k) and M1(x_k) is mapped into an "AND" logic and the multiplication of M2(c_k) and M2(x_k) is mapped into another "AND" logic. The bit-wise re-scaling operation is implemented by two RSM modules, respectively. Finally, the output is half of the sum of outputs of the two RSM modules.

4. Evaluation and Implementation

In this section, the SNR performance simulation results are firstly presented. Afterward, the hardware implementation is compared with the existing works.

4.1. Performance Simulation

Firstly, the SNR performance of stochastic logic-based multiplication unit under different bit stream length $N = 4, 8, 16, ..., 256$ is shown as Figure 10. Using the proposed adaptive scaling algorithm, the SNR of bipolar scheme [6] and two-line scheme [2] is significantly improved by 12 dB and 8 dB, respectively. Combining the Antithetic Variables method, the SNR is further improved by 6dB.

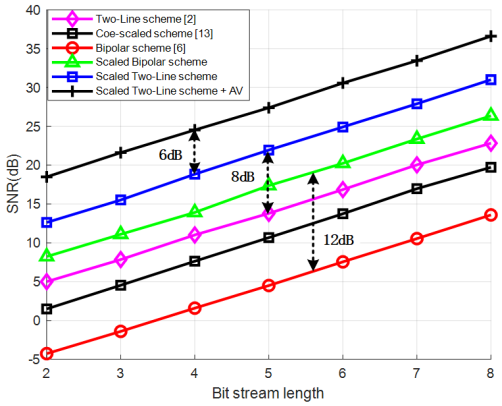

Figure 10. SNR performance evaluation on stochastic logic-based multiplication unit.

Afterward, the ASA based SFIR filter with 48-tap under different bit stream length $N = 2, 4, 8, 16, ..., 1024$ was simulated, and the results are shown in Figure 11a. The SNR performance gains on the bipolar scheme and two-line scheme are 33 dB and 17 dB, respectively. Furthermore, the ASA-AV-based SFIR filter gained 6 dB on SNR performance compared with the ASA-based scheme. The fix-point TCS-based scheme is also presented in Figure 11 as a comparison, which is optimized using state variable analysis method [18].

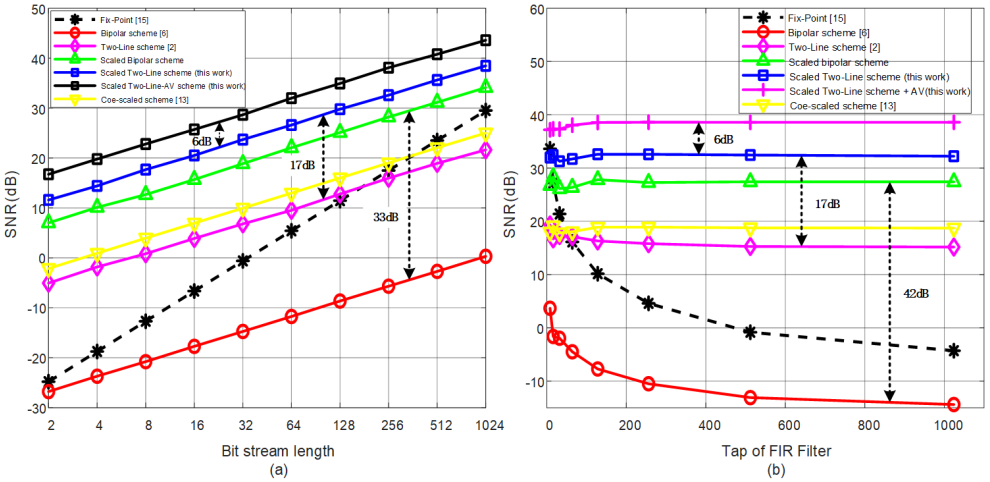

Figure 11. SNR performance evaluation on: (**a**) different bit stream lengths; (**b**) different filter taps.

Furthermore, the SNR performance of SFIR filter with bit stream length $N = 256$ under different taps are illustrated as Figure 11b, which indicates that the proposed ASA method and AV method both contribute stable accuracy gains with increasing filter taps.

Finally, the magnitude responses of 47-th order lowpass filer under different bit stream length is shown as Figure 12. The proposed ASA method significantly improves the computational accuracy: on the bipolar scheme and the two-line scheme, 33 dB improvement and 9 dB improvement are achieved, respectively, in the case of $N_{sto} = 2^{14}$. In addition, the ASA-AV scheme has 6 dB improvement compared with the ASA scheme.

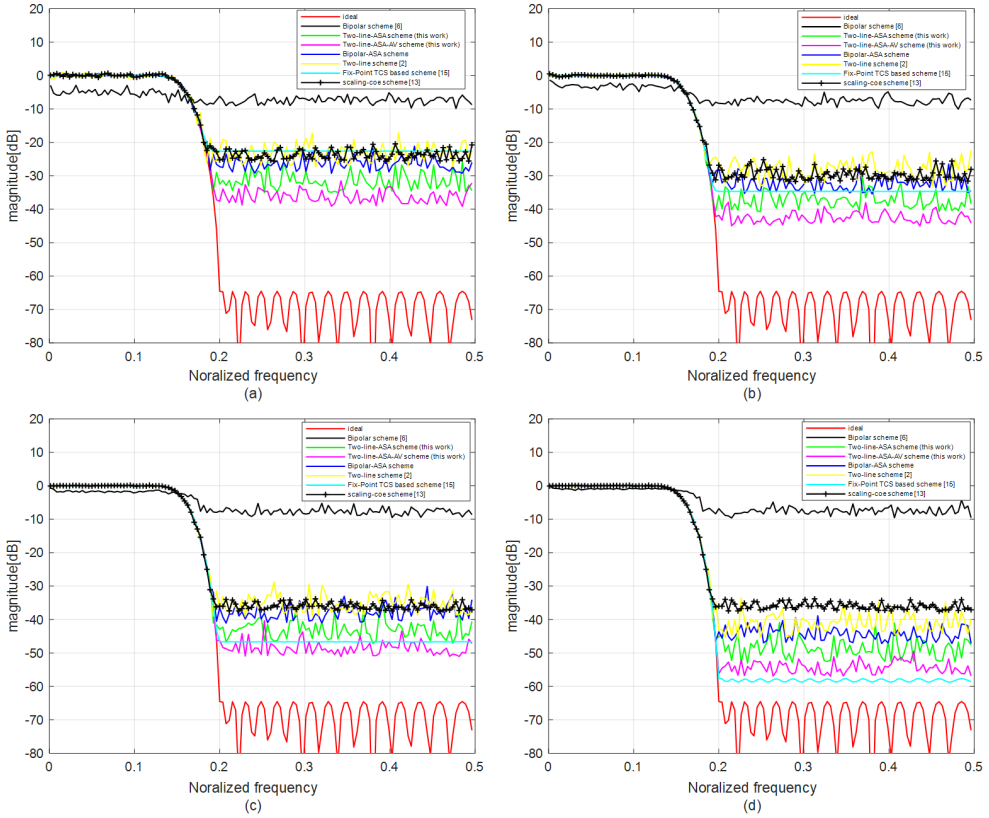

Figure 12. Magnitude responses of 47th-order SFIR filters: (**a**) $N_{sto} = 2^8$(Fix-Point 8 bits), (**b**) $N_{sto} = 2^{10}$(Fix-Point 10 bits), (**c**) $N_{sto} = 2^{12}$(Fix-Point 12 bits), and (**d**) $N_{sto} = 2^{14}$(Fix-Point 14 bits).

4.2. Hardware Implementation

The proposed AV–ASA–based SFIR is implemented using VHDL and synthesized with Synopsys design complier (DC) using the SMIC 90 nm library, shown as Table 1. The bipolar scheme [6] and two-line scheme [2] are also listed as a comparison with 64 filter taps and 256-length bit stream. Furthermore, the binary fix-point FIR filter is also synthesized with the same CMOS technology and listed in Table 1.

Table 1. Implementation results comparison.

SFIR Schemes	Bipolar SFIR [6]	Two-Line SFIR [2]	MUX SFIR [12]	Binary FIR (Fix-Point)	ASA-SFIR (This Work)	AV-ASA-SFIR (This Work)	
CMOS Tech.	90 nm	90 nm	90 nm *	90 nm	90 nm	90 nm	90 nm
Filter Tap	64	64	64	64	64	64	64
Bit Stream Length (bits)	256	256	256	–	256	256	16
Fix-Point Width (bits)	–	–	–	9	–	–	–
SNR (dB)	−4.41	17.21	–	23.51	31.38	37.40	25.64
Error ($\times 10^{-2}$)	34.42	2.8	2.38	1.5	0.39	0.2	0.85
Clock (MHz)	800	750	–	200	750	750	750
Area (um^2)	18,304	44,800	≈14,000 **	229,452	49,286	59,121	59,121
Power (mW)	–	–	–	2.40	1.913	2.31	2.31
Latency (ns)	320.00	341.32	–	5.00	341.32	341.32	21.33
Throughout (MSample/s)	3.13	2.92	–	200.00	2.92	2.92	46.88
Hardware Efficiency (MS/s/mm^2)	0.17	0.07	–	0.87	–	–	0.79

* Original 45 nm results is converted to 90 nm technology by $\times 4((90/45)^2)$ according to [19] for a fair comparison. ** Only 4, 8, 16, 32, 33, 100, 300-tap FIR filters are reported in [12], the area consumption of 64-tap filter is derived approximately.

Firstly, all of the stochastic filters take lower area cost compared with the binary filter due to the simple hardware architecture. The Bipolar, Two-Line, MUX, and AV-ASA schemes show 92%, 80%, 94%, and 74% area reduction compared to binary scheme, respectively. However, the processing latency is much higher than that of binary filter, which is caused by the long bit streams. Using the adaptive scaling algorithm and antithetic variable method, the proposed AV-ASA-SFIR scheme (16 bit stream length) achieves comparable hardware efficiency with respect to binary FIR filter, where the hardware efficiency is defined as throughput-to-area ratio as

$$\text{Hardware Efficiency}(\text{MS/s/mm}^2) = \frac{\text{Throughput}(\text{MS/s})}{\text{Area}(\text{mm}^2)}. \tag{19}$$

Among the SFIR schemes, the two-line scheme greatly improves the SNR performance from -4.41 dB to 17.21 dB, with 2.4 times chip area consumption overhead, compared with the bipolar scheme. The main hardware cost lies on the two-line stochastic adder (TSA) module. The proposed ASA–SFIR scheme gains 14.17 dB on SNR performance compared with the two-line scheme, with only 10.01% more hardware cost. Benefitting from the simple architecture of the SM and RSM module, the proposed design does not increase the critical path delay, achieving the same clock frequency as the two-line scheme. Moreover, combining the ASA and AV method, the AV–ASA–SFIR scheme gains 20.19 dB on SNR compared with the two-line scheme, with 32% area consumption overhead. Due to the significant improvement on accuracy, the proposed design require much shorter bit stream length compared with the existing stochastic FIR filters and shows advantages on processing latency. Note that the hardware architecture of the proposed design requires no change when modifying the bit stream length to achieve a trade-off between accuracy and latency.

4.3. Discussion

The representation noise analysis and the proposed adaptive scaling algorithm is based on a general stochastic bit stream and not specialized for the SFIR filter, which makes it possible to extend it to many other stochastic computing-based DSP systems, such as fast Fourier transform (FFT), discrete wavelet transform (DWT), and support vector machine (SVM). The proposed adaptive scaling method shall be considered in these modules in the future.

5. Conclusions

This paper presents a high-accuracy SFIR filter design based on an adaptive scaling algorithm. The relationship between representation noise and represented values of a stochastic bit stream is theoretically analyzed, providing a potential way to improve the accuracy under the same stochastic bit stream length. Afterward, an adaptive scaling algorithm (ASA) is proposed for the SFIR to scale the input signals into low-noise regions adaptively.According to the simulation results on a 64-tap FIR filter, the ASA and AV methods gained 17 dB and 6 dB in terms of signal-to-noise ratio (SNR), respectively. Finally, the hardware architecture of the proposed ASA-based SFIR (ASA-SFIR) is designed and implemented, which demonstrates 4.6 times hardware efficiency improvement with respect to the existing SFIR schemes.

Author Contributions: Conceptualization, J.H. and K.H.; methodology, Y.Z. (Ying Zhang), Y.Z. (Yubin Zhu) and K.H.; investigation, Y.Z. (Ying Zhang) and Y.Z. (Yubin Zhu); simulation, Y.Z. (Ying Zhang) and Y.Z. (Yubin Zhu); hardware, K.H.; writing, Y.Z. (Ying Zhang), K.H., J.H. and J.W. All authors have read and agreed to the published version of the manuscript.

Funding: National Natural Science Foundation of China under Grant No. 62001277 and 62001276. Guangdong Basic and Applied Basic Research Fund under Grant No. 2019A1515110560. National key research and development plan No. 2018YFB1801500.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Jiang, Q. FIR Filter Banks for Hexagonal Data Processing. *IEEE Trans. Image Process.* **2008**, *17*, 1512–1521. [CrossRef] [PubMed]
2. Yuan, B.; Wang, Y. High-accuracy FIR filter design using stochastic computing. In Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Pittsburgh, PA, USA, 11–13 July 2016; pp. 128–133.
3. Chen, J.; Hu, J.; Sobelman, G.E. Stochastic MIMO Detector Based on the Markov Chain Monte Carlo Algorithm. *IEEE Trans. Signal Process.* **2014**, *62*, 1454–1463. [CrossRef]
4. Alaghi, A.; Qian, W.; Hayes, J.P. The Promise and Challenge of Stochastic Computing. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2018**, *37*, 1515–1531. [CrossRef]
5. Liu, Y.; Venkataraman, H.; Zhang, Z.; Parhi, K.K. Machine learning classifiers using stochastic logic. In Proceedings of the IEEE 34th International Conference on Computer Design (ICCD), Scottsdale, AZ, USA, 2–5 October 2016; pp. 408–411.
6. Chang, Y.; Parhi, K.K. Architectures for digital filters using stochastic computing. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 2697–2701.
7. Koshita, S.; Onizawa, N.; Abe, M.; Hanyu, T.; Kawamata, M. Realization of FIR digital filters based on stochastic/binary hybrid computation. In Proceedings of the International Symposium on Multiple-Valued Logic, Sapporo, Japan, 18–20 May 2016; pp. 223–228.
8. Abbaszadeh, A.; Azerbaijan, A.; Sadeghipour, K.D. A new hardware efficient reconfigurable fir filter architecture suitable for FPGA applications. In Proceedings of the 17th DSP 2011 International Conference on Digital Signal Processing, Corfu, Greece, 6–8 July 2011; pp. 4–7.
9. Arash Ardakani, F.L.; Gross, W.J. Hardware implementation of FIR/IIR digital filters using integral stochastic computation. In Proceedings of the ICASSP 2016, Shanghai, China, 20–25 March 2016; pp. 6540–6544.
10. Chen, J.; Hu, J. A novel FIR filter based on stochastic logic. In Proceedings of the IEEE International Symposium on Circuits and Systems, Beijing, China, 19–23 May 2013; pp. 2050–2053.
11. Brown, B.D.; Card, H.C. Stochastic neural computation I: Computational elements. *IEEE Trans. Comput.* **2001**, *50*, 891–905. [CrossRef]
12. Ichihara, H.; Sugino, T.; Ishii, S.; Iwagaki, T.; Inoue, T. Compact and accurate digital filters based on stochastic computing. *IEEE Trans. Emerg. Top. Comput.* **2019**, *7*, 31–43. [CrossRef]
13. Kim, K.; Kim, J.; Yu, J.; Seo, J.; Lee, J.; Choi, K. Dynamic energy-accuracy trade-off using stochastic computing in deep neural networks. In Proceedings of the ACM Press the 53rd Annual Design Automation Conference, Austin, TX, USA, 5–9 June 2016; pp. 1–6.
14. Koshita, S.; Onizawa, N.; Abe, M.; Hanyu, T.; Kawamata, M. High-Accuracy and Area-Efficient Stochastic FIR Digital Filters Based on Hybrid Computation. *IEICE Trans. Inf. Syst.* **2017**, *100*, 1592–1602. [CrossRef]
15. Han, K.; Hu, J.; Chen, J.; Lu, H. A low complexity sparse code multiple access detector based on stochastic computing. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2017**, *65*, 769–782. [CrossRef]
16. Toral, S.L.; Quero, J.M.; Franquelo, L.G. Stochastic pulse coded arithmetic. In Proceedings of the IEEE International Symposium on Circuits and Systems, Geneva, Switzerland, 28–31 May 2000; Volume 3, p. 1.
17. Han, K.; Wang, J.; Gross, W.J.; Hu, J. Stochastic bit-wise iterative decoding of polar codes. *IEEE Trans. Signal Process.* **2018**, *67*, 1138–1151. [CrossRef]
18. Parhi, K.K. *VLSI Digital Signal Processing Systems: Design and Implementation*; John Wiley & Sons: Hoboken, NJ, USA, 2007.
19. Wong, C.C.; Chang, H.C. Reconfigurable turbo decoder with parallel architecture for 3GPP LTE system. *IEEE Trans. Circuits Syst. II Express Briefs* **2010**, *57*, 566–570. [CrossRef]