computation

MDPI

Article

Accuracy Analysis on Design of Stochastic Computing in Arithmetic Components and Combinational Circuit

P. Ashok and B. Bala Tripura Sundari *

Department of Electronics and Communication Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore 641112, India
* Correspondence: b_bala@cb.amrita.edu

Abstract: Stochastic circuits are used in applications that require low area and power consumption. The computing performed using these circuits is referred to as Stochastic computing (SC). The arithmetic operations in this computing can be realized using minimum logic circuits. The SC system allows a tradeoff of computational accuracy and area; thereby, the challenge in SC is improving the accuracy. The accuracy depends on the SC system's stochastic number generator (SNG) part. SNGs provide the appropriate stochastic input required for stochastic computation. Hence we explore the accuracy in SC for various arithmetic operations performed using stochastic computing with the help of logic circuits. The contributions in this paper are; first, we have performed stochastic computing for arithmetic components using two different SNGs. The SNGs considered are Linear Feed-back Shift Register (LFSR) -based traditional stochastic number generators and S-box-based stochastic number generators. Second, the arithmetic components are implemented in a combinational circuit for algebraic expression in the stochastic domain using two different SNGs. Third, computational analysis for stochastic arithmetic components and the stochastic algebraic equation has been conducted. Finally, accuracy analysis and measurement are performed between LFSR-based computation and S-box-based computation. The novel aspect of this work is the use of S-box-based SNG in the development of stochastic computing in arithmetic components. Also, the implementation of stochastic computing in the combinational circuit using the developed basic arithmetic components, and exploration of accuracy with respect to stochastic number generators used is presented.

Keywords: stochastic number generation; computing; arithmetic; accuracy; stochastic combinational circuit; error analysis; VLSI design

Citation: Ashok, P.; Bala Tripura Sundari, B. Accuracy Analysis on Design of Stochastic Computing in Arithmetic Components and Combinational Circuit. *Computation* **2023**, *11*, 237. https://doi.org/10.3390/computation11120237

Academic Editors: Michele Bonnin and Markus Kraft

Received: 21 September 2023
Revised: 6 November 2023
Accepted: 19 November 2023
Published: 1 December 2023

1. Introduction

Stochastic Computing (SC) is a new computing paradigm that has evolved as an alternative computing system that saves power and area consumption. In this type of computing, data are represented in terms of the probabilistic appearances of zeros and ones in a bit stream, referred to as stochastic number [1]. SC numbers are generally represented in unipolar, bipolar or inverted bipolar formats [2]. In this work, the unipolar format has been used because it simplifies the hardware implementation of stochastic computing circuits and typically utilizes fewer transistors leading to low power consumption. SC computation reduces processing time, memory requirement and the challenges of complex intensive computations when compared to conventional computing [3]. SC has been used for developing retiming signal processing filters using high level synthesis [4]. Lee et.al put forth possible hardware utilization of SC networks and a methodology for implementing an SC-based CNN using AI-based applications [5].

The design of the SC system comprises three blocks; the first block is the input to the stochastic system, termed as stochastic number generator (SNG), which converts a binary input to a stochastic bit stream; the second block is the stochastic computing core and SC-based design units required to build the SC system that performs stochastic operations

of the function required; the final block is the stochastic-to-binary converter to obtain the output in the binary form.

A new redundancy strategy based on approximate computing adder has been developed with a greater error-tolerance enabling an acceptable output quality that is an essential requirement in safety-critical case applications. Arithmetic functions with varying significance attached to their output bits are implemented and find varied applications that are inherently error-tolerant and that require higher reliability as safety-critical applications like space. These applications are digital signal processing, neuromorphic computing, big data mining, low-power graphics processing. Digital image, video, and audio processing are more error-tolerant since minor variations in an image or video frames or a feeble noise in audio may not be recognized by users due to human perception [6]. Although an SC-based computation saves in energy/power/area in terms of resources and computation time, the same is achieved with a trade off of accuracy.

The important parameter of efficiency is determined by the accuracy and is measured here in this work in terms of mean square error (MSE). This paper presents the accuracy analysis of basic arithmetic components implemented using two different stochastic number generators. Also, important SC-based designs have been implemented and followed up by analyzing and tabulating the results of accuracy in terms of MSE as the parameter of efficiency of an SC system.

- We have considered two different SNGs and carried out accuracy analysis for both namely LFSR-based SNG and S-box-based SNG. LFSR-based SNG is the traditional method used for generating stochastic numbers. S-box-based SNG is the new method used to develop the stochastic system performed here.
- Stochastic computing is implemented for basic arithmetic components using SC bit streams generated using LFSR-based SNG and S-box-based SNG.
- Also, an SC-based combinational circuit is realized using an algebraic expression for the two different SNGs.
- The output computed for all of the SC circuits is analyzed for accuracy measurement in terms of MSE.
- With the help of the computational analysis, accuracy is measured for the developed stochastic arithmetic component and the stochastic combinational circuit.

2. Related Works on Stochastic Computing Elements

Stochastic computing has found applications in image processing, pattern recognition, deep learning and AI. Based on the concept of statistically independent sequences as the SNG, a modulator-based system that generates binary sequences has been developed [7]. Considering that the SNG needs more area and power than SC computing cores, and also due to the importance of error measurement at the output, an error-cancelling conversion unit [8] and even-distribution coding method have been implemented [9]. The computational cost of the SNG in an SC system has been reduced by using a methodology of decomposition in the design and re-use of its functional unit [10]. Error reduction by register-based rearrangement techniques and sharing of the SNG's sub modules have been used to reduce the overhead cost in the design of the SNG [11]. Also, a parallel bit stream generator designed using a single clock unit has been experimented with for reducing the hardware cost of the SNG [12].

The various types of SC computing-based basic arithmetic components and their various stochastic computing formats have been implemented [13]. SC-based combinational and sequential circuits for stochastic generation units and the possible applications and various stochastic algorithms that can solve the challenges in stochastic system have been defined elaborately and implemented [14]. Division circuit in SC logic is developed and used in the CORDIV method, developed based on the understanding from the Gaines and ADDIE's basic ratio format [15]. An architecture for stochastic division has been developed for digital processing systems and also taking into account the correlation of SC streams for square root computations [16,17].

In addition to combinational circuits using SC methods, SC-based sequential circuits have also been developed. Stochastic computing had been implemented for complex systems like exponential and Tanh functions, using finite state machines and found to have much tolerance on soft-errors thus improving accuracy [18]. Architectures for complex computation in the digital domain have been constructed using approximation techniques that focus on exponential, Tanh, COS, and sinusoidal functions [19]. It has been reported that latency in computation is a concern along with overhead in area due to which insertion of flip flops has been used in stochastic circuits [20]. To perform normalization and increase the throughput, an efficient converging normalization unit had been used [21]. A deterministic stochastic system with down sampling has been introduced to overcome fluctuation and correlation to develop an accurate system [22]. An isolation-based method has been developed for handling the accuracy issue in an SC system. Errors were analyzed and mitigated by probability transfer metrics, thereby isolation and degeneration were discussed, and found to be having a cost advantage [23].

Thus stochastic computing (SC), originally proposed in the 1960s as a new methodology to be adapted in lieu of traditional binary computing, is an approximate computing technique that has been researched and has found increasing interest due to its advantages of reduced logic gates implementations. This enhances its ability to compress complex functions and fit into a lower number of logic gates. Such characteristic has motivated the development of different methods for the use of SC to implement Artificial Neural Network (ANNs) in hardware and also has found significant importance in the implementation of CNNs.

Stochastic CNNs have been facing difficulties in SC implementation and the challenges are the cost in terms of hardware resources required to implement different Random Number Generators (RNG), the precision degradation between layers produced by the lack of decorrelation between signals, and the implementation of a stochastic computing based activation function circuitry. Various CNN models have been developed such as hybrid stochastic binary CNN architectures, where the first convolutional layer uses stochastic computing, other layers and computations are implemented in binary conventional logic and only multiplication units are implemented in SC for fully parallel CNN hardware [24].

The most recent advances in SC computation are the extension of stochastic computing to neural networks pertaining to various AI-based applications, and a related work in this area is reported to be the implementation of stochastic neural computation using neural computational elements [25,26]. In the development of handling nonlinear activation functions required for CNN, a deep neural network focusing on area and parallel processing was been designed, with the aim of developing an accurate system with stable hardware resources [27]. In order to focus on correlation factor, an SC-based CNN has been proposed where the hardware resource has been saved with the aim of sustaining the accuracy [28]. The computation time for SC-based CNN using MAC unit has been proposed. An approach for binarized neural networks was developed for the Modified National Institute of Standards and Technology database(MNIST) and Canadian Institute For Advanced Research(CIFAR) data focusing on performance of the convolutional binarized network. This has been extended for ASIC implementations [29–32]. The authors have suggested the importance of a CNN accelerator and the use of Python as an Algorithmic High-Level Language, for the realization and the implementation and FPGA based accelerators for CNNs. They can be used to perform optimal selection (in the sense of multi-objective optimization) of the design parameters of an accelerator for trade-off study and resource minimization. This would be a prominent background for the future work for high level synthesis of CNN layers [33].

The main focus of our research work is to build an efficient SC system focusing on accuracy improvement at all levels till the development of the CNN system. As a first step towards the same an S-Box-based the SNG methodology, (where-in the S-Box [34,35] in cryptographic algorithm) is used as the stochastic bit stream generator for building up the

SC components. The developed components will be used for the development of CNN layers.

Hence, the SC circuits need to be thoroughly investigated for errors in computation for the different SNGs methods used. Therefore, the organization and contribution in this paper are briefed as follows:

Section 3.1 describes the stochastic number generation where the two SNGs are used LFSR-based SNG and S-box-based SNG. The contribution in the SNG part is the implementation of the SC-based arithmetic components using an S-Box-based SNG.

Section 3.2 describes the implementation of stochastic computing in the basic arithmetic components. In this, performance is elaborated across the 4-bit length and 8-bit lengths to analyze the accuracy. Both SNGs are used as input for computing arithmetic components, from which a comparison will be made for accuracy analysis.

Section 3.3 describes the implementation of stochastic computing to develop combinational circuit. The input considered here is both SNGs with which the accuracy verification will be conducted. The combinational circuit is newly designed by solving the algebraic expression. This is developed by incorporating the basic arithmetic components to form a combinational circuit.

Section 4 presents the analysis of computation based on the error calculation. Accuracy has been calculated from the error analysis.

Sections 4.2 and 4.3 depicts the accuracy analysis for basic arithmetic components and combinational circuits considering all the bit lengths involved in the implementation.

3. Proposed Computing Methodology

The methodology is subdivided into three parts namely, stochastic number generation, stochastic arithmetic computation, and stochastic combinational circuit. This is followed by error analysis and accuracy measure. The block representation of the process involved in the proposed methodology is depicted below in Figure 1.

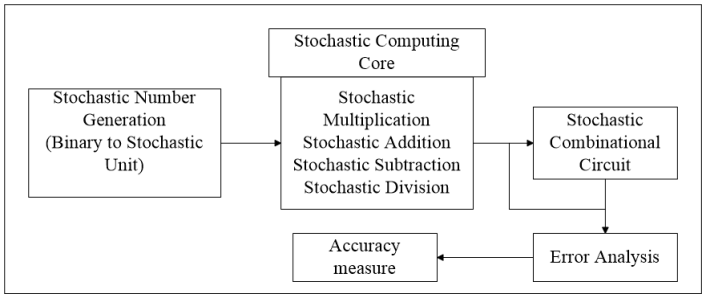

Figure 1. Block diagram of the proposed stochastic system.

3.1. Stochastic Number Generation

Stochastic number generation is the main block of any stochastic system and is used in conversion of binary input data to stochastic data. In this work, we have considered two different stochastic number generation methods.

3.1.1. LFSR-Based SNG—Algorithm

Linear Feedback Shift register based the SNG is a traditional way of generating random numbers as shown in Figure 2. The random values obtained from the register will be sent to a comparator. It compares the first number and second number, if the second number is larger than the first, it will generate the bit value '1' else it will generate a bit value of '0'. In this way, the LFSR-based SNG is used as a generator in the design. The LFSR-based SNG has to be repeated for each input unit for the Stochastic number to be generated.

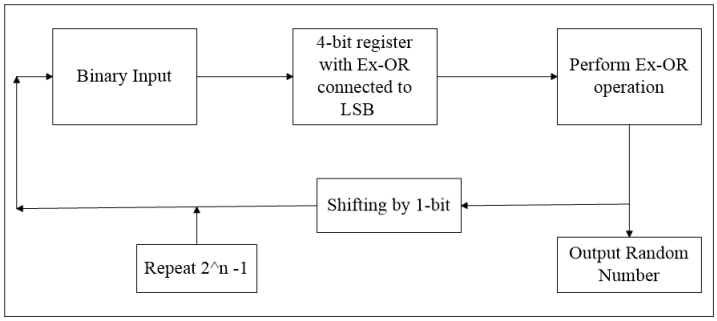

Figure 2. Block diagram of LFSR-based SNG.

3.1.2. S-Box-Based SNG Algorithm

The S-box-based SNG Figure 3 is a new method used for stochastic number generation. The method generates a random number which will be used for multiple inputs without repeating the functionality for each SC stream to be generated. So a single S-box SNG will serve as an SNG source for many computing units. The number generated from the SNG could be directly used as a stochastic bit for computation. The number obtained from this SNG itself is a stochastic number that could be used as an input for computing units. Components used for converting binary to stochastic are appropriate to be used as an SNG.

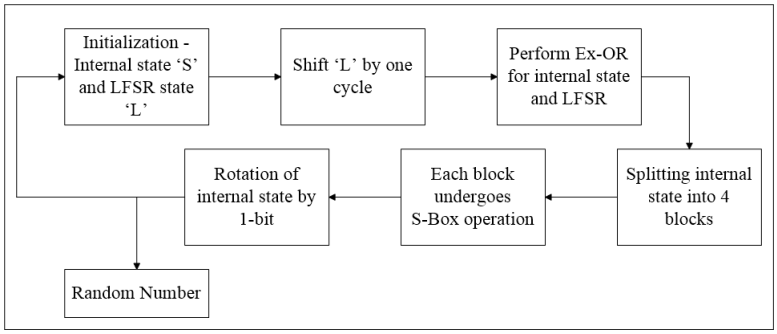

Figure 3. Block diagram of S-Box-based SNG.

3.2. Implementation of Stochastic Computing in Arithmetic Operation

The second part of the proposed methodology is the implementation of stochastic computing in arithmetic operations like multiplication, addition, subtraction, and division. In these operations, two SNGs are used for the input of the computing system.

In this work, stochastic computing is performed for basic arithmetic operations using appropriate components. The computations to be performed for the basic components require stochastic number generators. Hence, we use two different stochastic number generation units for all the arithmetic operations. So, the performance of stochastic computing with respect to these SNGs has been conducted by implementing both SNGs in the arithmetic operations. This will stabilize the performance of the SC system with respect to accuracy. The construction of these components are conducted using basic logic elements like logical AND gate, logic multiplexer and storage flip flop.

The binary input is initially taken and the same is converted to stochastic form using two different SNGs to perform computation in the following mentioned components and the results are analyzed for accuracy and error.

3.2.1. Stochastic Multiplication

The logical AND gate is used for implementing multiplication operations in stochastic computing. The inputs of AND gate are chosen as two-bit streams 'X' and 'Y', respectively, that are stochastic. The output is 'Z'. The inputs of the multiplier (AND gate) must be statistically independent. The stochastic multiplication operation is performed to overcome multiplication unwrapping or partial product generation. Considering two inputs AND

gate having X and Y as input, mathematically stochastic multiplication is expressed as the probability of (X = 1) multiplied by the probability of (Y = 1), producing a probability of (Z = 1).

3.2.2. Stochastic Addition

The logical multiplexer is used to perform stochastic addition operations. In the stochastic context, the inputs are considered as probabilities, and hence on adding any two inputs, the resultant does not lie within (0, 1) limits. So scaled addition is performed with the help of a multiplexer. The two input bit streams are given at the input port X and Y, whereas select input is given as constant 0.5 because the output should have its value between [0,1] and treated as a probability. This is how floating ranges are scaled; this kind is referred to as scaled addition. The advantage is that there is no need for the carry element to propagate from bit to bit. Mathematically, stochastic addition is written as, $P(Z = 1) = P(S = 1).P(A = 1) + P(1 - S = 1).P(B = 1)$, where A, B are the two inputs, S is the select line and Z is the output of the multiplexer. The scaling parameter can be expressed as Z = 0.5 (A+B).

3.2.3. Stochastic Subtraction

The Logical multiplexer with one of its inputs connected to the NOT gate is used to perform stochastic subtraction operations. The two input bit streams are given at the input ports X and Y, whereas the select input is given as constant 0.5 because the output should have a value between [0,1]. In terms of probabilistic means, stochastic subtraction is expressed as the Probability of (Z = 1) if the product of P (S = 1) and P (A = 1) & sum of the product of Probability of $(1 - S = 1)$ and Probability of $(1 - B = 1)$.

3.2.4. Stochastic Division

Stochastic division takes the Jack Kilby(JK) flip flop(FF) where the FF sets states 0 and 1 for 01 and 10, respectively; eventually the 00 state is not unchanged whereas if the FF is toggled. With this note, considering J & K to be the two inputs and mapping it to the probability concept as P_{x1} and P_{x2}, the output probability is generalized as $P_z = P_{x1}/(P_{x1} + P_{x2})$. This is normalized as $P_z = P_{x1}/P_{x2}$ with the condition of P_{x1} to be minimal. To implement division unipolar format is used. Mathematically, the division is carried out based on the following rule, $P(Z = 1) = (P_{x1} = 1)/(P_{x1} + P_{x2}) = 1$.

3.3. Stochastic Implementation of Combinational Circuit

In this section, the idea is to develop an algebraic expression and implement stochastic computing using the basic arithmetic components designed above for the expression considered. The structure developed uses both SNGs as the input for the system separately and performs the accuracy measure for the combinational circuit system.

Consider the algebraic expression for developing a combinational circuit $A^2 - B^2$. In order to implement the expression in the stochastic domain, the expression can be written as, $F = (A + B)(A - B)$

The algebraic expression can be incorporated into a stochastic system using the stochastic components addition, subtraction, and multiplication. The structure of the expression is depicted below in Figure 4.

The expression is incorporated in the above diagram, where the input is binary for all number generator units. The input binary is converted to a stochastic number using a stochastic number generator. The traditional LFSR-based SNG and the proposed S-box-based SNG are the SNGs used separately to convert binary inputs. After converting, the first part is the realization of A + B. This is accomplished by using a multiplexer with the help of stochastic streams. In parallel, A − B is realized using a multiplexer in which one of the inputs is negated to activate subtraction. The select input in both the components is due to the scaling of computed values between (0, 1). Finally, these two functions are combined

and realized using AND gate to perform multiplication. The resultant is the value obtained from the realization of the expression $A^2 - B^2$.

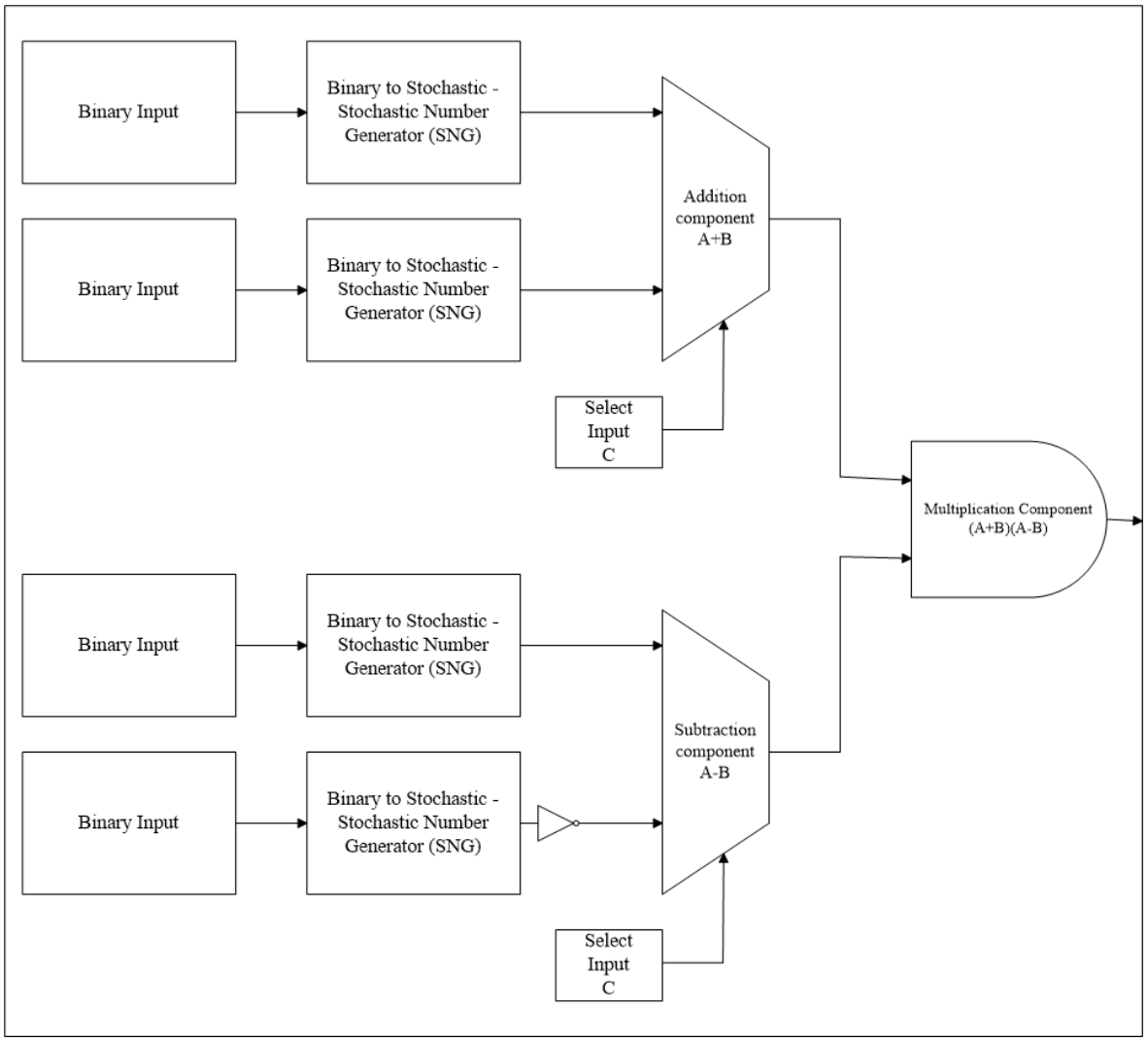

Figure 4. Stochastic combinational circuit.

4. Experimental Analysis and Inference

In this work, four scenarios have been considered and the same has been depicted in the block diagram. The aim of the work is to analyze the accuracy of the stochastic system that has been developed. The stochastic system considered here requires a stochastic number generator, and a stochastic computing component.

With these, the system is developed in two ways, first, two different stochastic number generators are used as the SNG for stochastic computing core (basic arithmetic components). So this stochastic system is analyzed for accuracy through error analysis from the SNGs considered and components. The second part is the use of these two SNGs and the developed basic components for the realization of an algebraic expression to implement in the stochastic computing logic. This part of the system is separately analyzed for accuracy through error measurement.

The first scenario is the development of an appropriate SNG to be used for computing purposes. The two SNGs used here are the LFSR-based SNG and S-Box-based SNG. These are used as SNGs for all the computing process separately. These are the primary vital parts for the efficient development of a stochastic system.

The second scenario considered here is the stochastic computing in basic arithmetic operations namely SC-based multiplication, addition, subtraction and division using the above-mentioned SNGs. Four arithmetic components are considered in order to analyze the stochastic system based on its operation since different logic elements are used for each operation. It is also conducted to emphasize that stochastic computing could be conducted by basic logic elements.

The third scenario is the realization of algebraic expression and the implementation of the same using stochastic computing. This is realized by basic arithmetic components that are mentioned earlier. Thus, this realization involves basic components and SNGs developed for conversion purposes. The entire part will be undergo accuracy measurement.

The fourth scenario is the accuracy measure conducted by considering all the scenarios mentioned above. Error analysis is conducted for all cases by identifying the difference in original value to the traditional value of the stochastic bit generated. This is conducted by considering the binary equivalent of the stochastic number which is used for determining the error difference.

4.1. Function Generation and Conditions

In the LFSR-based method, the experiment is carried with a 4-bit binary number as the input. Any 4-bit binary entered in the 4-bit register is considered. Ex-oR operation is performed for the two Least Significant Bit (LSB) of the register. The resultant value is the first output random value and the same is passed as input to the register, thereby the values in the register are shifted to the right. The process continues until it reaches $2^n - 1$ criteria. Hence for the work conducted, 15 values will be obtained after the cycle completes. This is referred to as a random number sequence and will be used as an input value for converting into a stochastic number.

Now, the random number obtained is compared with a random binary number using a comparator. The random number obtained from the LFSR is considered as 'A' and the binary number is considered as 'B'. The comparator is processed with the condition values in the 'A' should be less than the values in the 'B' to produce output as value '1'. This value is the required stochastic number.

The S-Box-based method has three steps. The first step is the initialization where the structure comprises the internal state 'S' and LFSR 'L'. It should be noted that the length of S and L should be the same. The internal state must be log(n) bits to the generated stochastic number. The second step is the number generation, the values in L are advanced by cycle one to perform Ex-OR with the internal state. The internal state is split into blocks of four bits. Each block is performed with the S-box circuit comprising of four different expressions as follows, where i_o, i_1, i_2 and i_3 are the input values of the internal state.

$Output1 = \bar{i}_2 \cdot (i_3) + (i_0 \cdot \bar{i}_1 \cdot \bar{i}_3) + (i_0 \cdot i_1 \cdot i_2)$
$Output2 = (i_0 \cdot \bar{i}_1 \cdot i_2 \cdot \bar{i}_3) + (i_1 \cdot \bar{i}_2) + (\bar{i}_0 \cdot \bar{i}_2 \cdot \bar{i}_3) + (i_o \cdot \bar{i}_2 \cdot i_3) + (\bar{i}_0 \cdot i_1 \cdot \bar{i}_3)$
$Output3 = (\bar{i}_0 \cdot i_1 \cdot \bar{i}_2 \cdot i_3) + (\bar{i}_1 \cdot \bar{i}_3) + (i_2 \cdot \bar{i}_3) + (i_0 \cdot \bar{i}_1 \cdot i_2)$
$Output4 = (i_o \cdot \bar{i}_1 \cdot \bar{i}_2) + (\bar{i}_0 \cdot i_1 \cdot \bar{i}_3) + (\bar{i}_1 \cdot i_3) + (\bar{i}_0 \cdot \bar{i}_2 \cdot i_3)$

The result obtained is rotated and is the required stochastic number. The same is further Ex-oR with LFSR in the next cycle.

The accuracy measurement and error analysis of the conventional LFSR and developed S-Box bit stream generation method are compared. The two methods are used in the implementations of arithmetic components and combinational circuits for an algebraic expression. The experimental computational results have been performed and detailed accuracy measure that includes analysis from computation followed by error calculation.

The comparison for conventional SC generation and S-Box-based SC is conducted in all the above to determine which method is better in accuracy and hence leading to error minimization which is the main focus in our work.

4.2. Experimental Results of Stochastic Arithmetic Components

4.2.1. Error Analysis for Stochastic Arithmetic Components

The error analysis table represents the error values of each computing component of 4-bit and 8-bit computations. The table comprises LFSR-based computing and S-Box-based computing in which the stochastic output and actual binary input of the sample have been shown for error calculation. The same has been performed and tabulated for stochastic combinational circuit design.

Tables 1 and 2 represent the error calculation for the computational values obtained from 4-bit and 8-bit lengths computation for various stochastic arithmetic components, the values have considerable differences in the two SNGs' performance for multiplication. It has been found that the S-Box-based implementation gives better accuracy and reduced error.

The following sections provide comparison of the two SC-based techniques used in arithmetic components and combinational circuits. A comparison is conducted for the accuracy and error analysis for the 4-bit implementations of the above type and then scaled up to 8-bit implementations.

Table 1. Error analysis of stochastic arithmetic computation—4 bit.

Stochastic Output (A)	Actual Output (B)	ERROR	Stochastic Output (A)	Actual Output (B)	ERROR
Stochastic Multiplication					
LFSR-based stochastic multiplication—4 bit			**S-Box-based stochastic multiplication—4 bit**		
2	168	160.09	2	110	106.036
1	96	94.010	2	77	73.05
1	80	78.01	0	48	48
1	64	62.01	1	24	22.04
2	35	31.11	1	20	18.05
1	20	18.05	0	8	8
Stochastic Addition					
LFSR-based stochastic addition—4 bit			**S-Box-based stochastic addition—4 bit**		
2	26	22.15	4	29	21.55
1	20	18.05	2	21	17.19
2	18	14.22	2	16	12.25
2	17	13.23	1	14	12.07
1	12	10.083	1	10	8.1
2	12	8.33	1	9	7.11
Stochastic Subtraction					
LFSR-based stochastic subtraction—4 bit			**S-Box-based stochastic subtraction—4 bit**		
1	11	9.09	1	10	8.1
3	1	4	2	7	3.57
1	4	2.25	1	4	2.25
4	2	2	2	1	1
2	10	1.6	1	2	0.5
2	4	1	2	3	0.33

Table 1. *Cont.*

Stochastic Output (A)	Actual Output (B)	ERROR	Stochastic Output (A)	Actual Output (B)	ERROR
Stochastic Division					
LFSR-based stochastic division—4 bit			**S-Box-based stochastic division—4 bit**		
2	0.125	28.125	1	0.0714	12.077
2	0.166	20.262	3	0.55	10.91
2	0.25	12.25	1	0.0833	10.088
2	0.333	8.345	1	0.1	8.1
2	0.375	7.041	2	0.636	2.925
3	12	6.75	2	0.75	2.083

Table 2. Error analysis of stochastic arithmetic computation—8 bit.

Stochastic Output (A)	Actual Output (B)	ERROR	Stochastic Output (A)	Actual Output (B)	ERROR
Stochastic Multiplication					
LFSR-based stochastic multiplication—8 bit			**S-Box-based stochastic multiplication—8 bit**		
3	46,256	46,253	2	39,872	39,868
3	37,636	37,630	2	37,286	37,282
3	28,836	28,830	2	11,685	11,681
3	17,577	17,571	1	11,534	11,532
3	11,800	11,794	1	10,251	10,249
2	5313	5309	1	4648	4646
Stochastic Addition					
LFSR-based stochastic addition—8 bit			**S-Box-based stochastic addition—8 bit**		
4	432	424.03	3	402	396.02
3	388	382.02	4	387	379.04
4	340	332.04	5	262	252.09
4	298	290.05	1	252	250
5	218	208.11	4	207	199.07
2	194	190.02	4	194	186.08
Stochastic Subtraction					
LFSR-based stochastic subtraction—8 bit			**S-Box-based stochastic subtraction—8 bit**		
6	136	124.26	1	107	105
5	132	122.18	5	106	96.23
4	128	120.12	3	85	79.10
3	65	59.13	4	46	38.34
5	65	55.38	4	25	17.64
8	40	25.6	4	23	15.69

Table 2. *Cont.*

Stochastic Output (A)	Actual Output (B)	ERROR	Stochastic Output (A)	Actual Output (B)	ERROR
Stochastic Division					
LFSR-based stochastic division—8 bit			**S-Box-based stochastic division—8 bit**		
5	0.086	280.32	6	182	170.19
3	97	91.09	3	0.1686	47.53
4	0.2857	48.28	6	0.8437	31.5
5	0.625	30.625	5	0.7946	22.252
6	33.66	22.729	6	1.101	21.79
5	1.18	12.36	4	0.8786	11.089

4.2.2. Accuracy Measure for Stochastic Arithmetic Components

The accuracy measure for stochastic arithmetic components (stochastic multiplication, stochastic addition, stochastic subtraction, stochastic division) for the two SNGs across 4-bit and 8-bit lengths is depicted in Figure 5, Figure 6, Figure 7 and Figure 8, respectively. In the accuracy graph, the X-axis represents the number of samples on computation and the Y-axis represents the error values of each computation. It is inferred that 4-bit computation of stochastic multiplication shows good performance differences between LFSR-based SNG and S-Box-based SNG from Figure 5. However, across all bit lengths, S-box SNG gives reduced error in computation and hence, is more accurate.

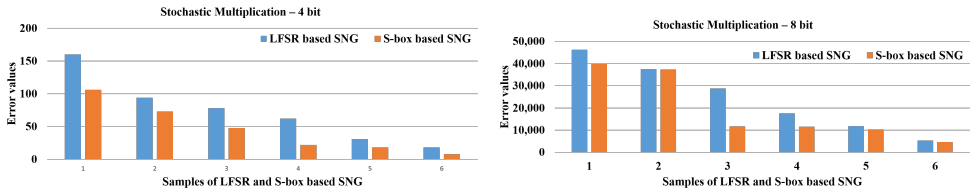

Figure 5. Error analysis in Stochastic multiplication.

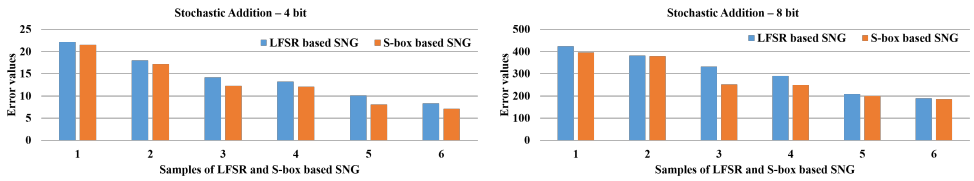

Figure 6. Error analysis in Stochastic addition.

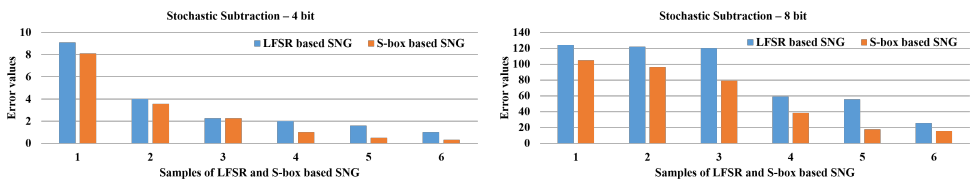

Figure 7. Error analysis in Stochastic subtraction.

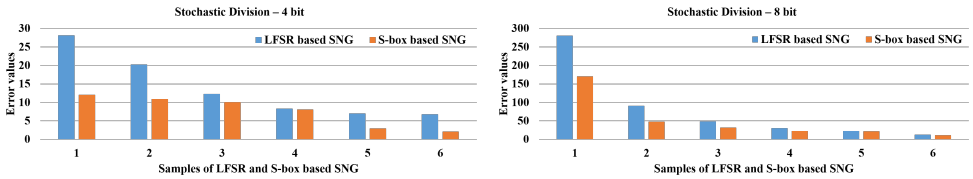

Figure 8. Error analysis in Stochastic division.

Figure 6 details the accuracy of stochastic addition performed with LFSR-based SNG and S-Box-based SNG. It is inferred that S-Box-based SNG works well for stochastic addition showing that this type of SNG has less errors when performing stochastic computation.

Accuracy is determined from error analysis and depicted in Figure 7 for stochastic subtraction. S-Box-based shows less error compared to LFSR-based SNG with respect to stochastic subtraction. This computation stays better for accurate stochastic components.

The accuracy measure is depicted in Figure 8 showing the performance of SNG's with respect to stochastic division. It is inferred that the computation of stochastic division shows good performance differences between LFSR-based SNG and S-Box-based SNG.

4.3. Experimental Results of Stochastic Combinational Circuit

4.3.1. Error Analysis for Stochastic Combinational Circuit

A stochastic computing based combinational circuit has been constructed and computing has been performed for the same. The computational results pertaining to LFSR-based SNG and S-Box-based SNG are used for determining accuracy and error calculation, respectively.

The error analysis for stochastic combinational circuit has been mentioned in the Table 3. The involvement of basic arithmetic components in this construction has shown a lower error rate for S-Box-based computation compared to traditional LFSR-based SNG.

Table 3. Error analysis of stochastic combinational circuit.

Stochastic Output (A)	Actual Output (B)	ERROR	Stochastic Output (A)	Actual Output (B)	ERROR
LFSR-based stochastic combinational circuit—4 bit			**S-Box-based stochastic combinational circuit 4 bit**		
1	96	94.0104	2	77	73.051
1	80	78.012	1	48	46.020
1	72	70.0138	0	45	45
1	64	62.0156	0	24	24
0	36	36	0	15	15
LFSR-based stochastic combinational circuit 8 bit			**S-Box-based stochastic combinational circuit 8 bit**		
2	5120	5116	1	4648	4646
1	5313	5311	3	3456	3450
2	3636	3632	1	1043	1041
3	2960	2954	2	1035	1031
1	2744	2742	5	182	172.13

4.3.2. Accuracy Measure for Stochastic Combinational Circuit

It is inferred from Figure 9 that the S-Box-based SNG is accurate with respect to the stochastic combinational circuit, as it is less prone to error in comparison with the LFSR-based SNG. Hence, the basic stochastic arithmetic components also evidence that the S-Box-based method is better for constructing an accurate stochastic system.

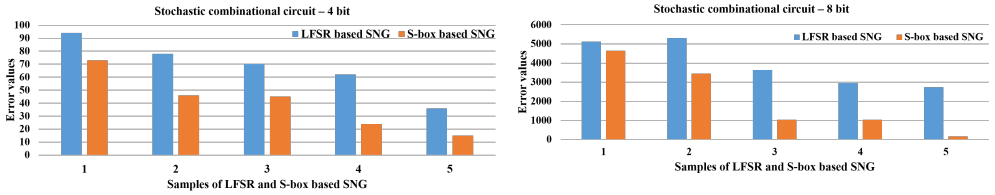

Figure 9. Error analysis in Stochastic combinational circuit.

4.4. Difference in Performance with Respect to SNG

The accuracy analysis could be performed by analyzing the error from the above computation, and its table is discussed below. The error measurement is conducted to determine the accuracy of the stochastic system for the stochastic number generator. The error calculation has been performed for the arithmetic components shown and combinational circuit. This is attempted for 10 calculations. Sample calculation of error is shown below.

Consider stochastic addition; the computation is conducted for a 4-bit bit stream. The first input is 0111, the second input is 0101, and the stochastic output is 0101. The corresponding values of the input are 7 and 5, respectively. The traditional addition value is 12. The stochastic output value is 2, as the number of 1's in the output sequence is considered as per the condition. So the error is calculated as the difference in the obtained value with the original value divided by the original, which gives the error of 8.33 in this case. In this way, the error is calculated for every value, and error analysis is conducted to measure the accuracy for LFSR and S-box-based SNG, respectively.

Table 4 summarizes the performance of stochastic computing in various arithmetic components. Accuracy can be determined from the table and it is inferred that the S-Box-based SNG method shows better in accuracy. While considering the bit length of various components, 4 bit length and 8 bit length sequences are considerably equal in accuracy determination.

Table 4. Accuracy comparison of various stochastic arithmetic components.

S. No	Component	LFSR Based SNG	S-Box Based SNG	Accuracy
1.	Stochastic Multiplication 4 bit	73.88	45.86	S-box stochastic multiplication Length—4 bit
	Stochastic Multiplication 8 bit	24564.5	19209.66	
2.	Stochastic Addition 4 bit	14.34	13.045	S-box stochastic addition Length—8 bit
	Stochastic Addition 8 bit	304.378	277.05	
3.	Stochastic Subtraction 4 bit	3.323	2.625	S-box stochastic subtraction Length—8 bit
	Stochastic Subtraction 8 bit	84.445	58.666	
4.	Stochastic Division 4 bit	13.795	73697	S-box stochastic division Length—4 bit
	Stochastic Division 8 bit	80.9006	50.725	
5.	Stochastic Combinational circuit 4 bit	68.01	40.6142	S-box stochastic combinational system 4 bit and 8 bit
	Stochastic Combinational circuit 8 bit	3951	2068.026	

5. Conclusions

In this paper, two different stochastic number generator methods are used to implement basic arithmetic components and a combinational arithmetic circuit for an algebraic expression. The above SC-based arithmetic components and stochastic combinational circuits for 4 and 8 bit streams are analyzed by error measurement for both types of SNGs namely LFSR-based SNG and S-box-based SNG to compare for accuracy. It is inferred from the analysis that the S-Box-based SNG method shows lower error across all bit streams—4 and 8 bit for SC-based arithmetic and combinational circuits developed.

The S-box SNG method will be used in the development of SC-based CNN, and the conventional LFSR will also be implemented for comparison. This will allow the design space exploration of the scaling of CNN layers to conduct trade-off analysis of accuracy versus power/energy scaling, computational time and hardware/resource utilization using the different SNGs used here.

6. Future Scope

The future scope of this work will be to determine the area overhead of the respective SNGs and design space exploration. Also, the measure of randomness of the SC data stream generated by the SNGs will be attempted by measuring the correlation of the bit stream generated by each. The increase in reliability can be attempted by bringing the redundancy of the architecture by appropriate parallelism in the implementation.

Author Contributions: Conceptualization, P.A.; Methodology, P.A.; Validation, P.A. and B.B.T.S.; Investigation, P.A.; Writing—original draft, P.A.; Writing—review editing, P.A. and B.B.T.S.; Supervision, B.B.T.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are available within the manuscript prepared.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, Y.; Liu, S.; Wang, Y.; Lombardi, F.; Han, J. A survey of stochastic computing neural networks for machine learning applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 2809–2824. [CrossRef] [PubMed]
2. Gaines, B.R. Origins of stochastic computing. In *Stochastic Computing: Techniques and Applications*; Springer: Cham, Switzerland, 2019; pp. 13–37.
3. Winstead, C. Tutorial on Stochastic Computing. In *Stochastic Computing: Techniques and Applications*; Springer: Cham, Switzerland, 2019; pp. 39–76.
4. Krishnapriya, P.N.; Bala Tripura Sundari, B. High level synthesis for retiming stochastic VLSI signal processing architectures. *Procedia Comput. Sci.* **2019**, *143*, 10–19.
5. Lee, Y.Y.; Halim, Z.A. Stochastic computing in convolutional neural network implementation: A review. *PeerJ Comput. Sci.* **2020**, *6*, e309. [CrossRef] [PubMed]
6. Balasubramanian, P.; Maskell, D.L.; Prasad, K. RESAC: A redundancy strategy involving approximate computing for error-tolerant applications. *Microelectron. Reliab.* **2023**, *150* , 115198. [CrossRef]
7. Neugebauer, F.; Polian, I.; Hayes, J.P. Framework for quantifying and managing accuracy in stochastic circuit design. *ACM J. Emerg. Technol. Comput. Syst.* **2018**, *14*, 1–21. [CrossRef]
8. Yang, M.; Hayes, J.P.; Fan, D.; Qian, W. Design of accurate stochastic number generators with noisy emerging devices for stochastic computing. In Proceedings of the 2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Irvine, CA, USA, 13–16 November 2017; pp. 638–644.
9. Zhakatayev, A.; Kim, K.; Choi, K.; Lee, J. An efficient and accurate stochastic number generator using even-distribution coding. *IEEE Trans.-Comput. Aided Des. Integr. Circuits Syst.* **2018**, *37*, 3056–3066. [CrossRef]
10. Karadeniz, M.B.; Altun, M . Sampling based random number generator for stochastic computing. In Proceedings of the 2017 24th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Batumi, Georgia, 5–8 December 2017; pp. 227–230.
11. Sakamoto, Y.; Yamashita, S. Efficient Methods to Generate Constant SNs with Considering Trade-off between Error and Overhead and Its Evaluation. *IEICE Trans. Inf. Syst.* **2020**, *103*, 321–328. [CrossRef]
12. Zhang, Y.; Wang, R.; Zhang, X.; Zhang, Z.; Song, J.; Zhang, Z.; Wang, Y.; Huang, R. A parallel bitstream generator for stochastic computing. In Proceedings of the 2019 Silicon Nano Electronics Workshop (SNW), Kyoto, Japan, 9–10 June 2019; pp. 1–2.

13. Akhtar, R.; Khanday, F.A. Stochastic Computing: Systems, Applications, Challenges and Solutions. In Proceedings of the 2018 3rd International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 15–16 October 2018; pp. 722–727.
14. Alaghi, A.; Qian, W.; Hayes, J.P. The promise and challenge of stochastic computing. *IEEE Trans. -Comput. Aided Des. Integr. Circuits Syst.* **2017**, *37*, 1515–1531. [CrossRef]
15. Chen, T.-H.; Hayes, J.P. Design of division circuits for stochastic computing. In Proceedings of the 2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Pittsburgh, PA, USA, 11–13 July 2016; pp. 116–121.
16. Temenos, N.; Sotiriadis, P.P. A New Technique for Stochastic Division in Unipolar Format. In Proceedings of the 2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST), Thessaloniki, Greece, 13–15 May 2019; pp. 1–4.
17. Wu, D.; Miguel, J.S. In-stream stochastic division and square root via correlation. In Proceedings of the 2019 56th ACM/IEEE Design Automation Conference (DAC), Las Vegas, NV, USA, 2–6 June 2019; pp. 1–6.
18. Najafi, M.H.; Li, P.; Lilja, D.J.; Qian, W.; Bazargan, K.; Riedel, M. A reconfigurable architecture with sequential logic-based stochastic computing. *ACM J. Emerg. Technol. Comput. Syst.* **2017**, *13*, 1–28. [CrossRef]
19. Luong, T.-K.; Nguyen, V.-T.; Nguyen, A.-T.; Popovici, E. Efficient architectures and implementation of arithmetic functions approximation based stochastic computing. In Proceedings of the 2019 IEEE 30th International Conference on Application—Specific Systems, Architectures and Processors (ASAP), New York, NY, USA, 15–17 June 2019; Volume 2160, pp. 281–287.
20. Li, Z.; Chen, Z.; Zhang, Y.; Huang, Z.; Qian, W. Simultaneous Area and Latency Optimization for Stochastic Circuits by D Flip-Flop Insertion. *IEEE Trans.-Comput. Aided Des. Integr. Circuits Syst.* **2018**, *38*, 1251–1264. [CrossRef]
21. Han, K.; Hu, J.; Chen, J.; Zhang, Z.; Lu, H. A fast converging normalization unit for stochastic computing. *IEEE Trans. Circuits Syst. II Express Briefs* **2017**, *65*, 501–505. [CrossRef]
22. Najafi, M.H.; Lilja, D. High quality down-sampling for deterministic approaches to stochastic computing. *IEEE Trans. Emerg. Top. Comput.* **2018**, *9*, 7–14. [CrossRef]
23. Ting, P.-S.; Hayes, J.P. Isolation-based decorrelation of stochastic circuits. In Proceedings of the 2016 IEEE 34th International Conference on Computer Design (ICCD), Scottsdale, AZ, USA, 2–5 October 2016; pp. 88–95.
24. Frasser, C.F.; Linares-Serrano, P.; de Los Rios, I.D.; Moran, A.; Skibinsky-Gitlin, E.S.; Font-Rossello, J.; Canals, V.; Roca, M.; Serrano-Gotarredona, T.; Rossello, J.L. Fully parallel stochastic computing hardware implementation of convolutional neural networks for edge computing applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**. [CrossRef] [PubMed]
25. Misra, S.; Bland, L.C.; Cardwell, S.G.; Incorvia, J.A.C.; James, C.D.; Kent, A.D.; Schuman, C.D.; Smith, J.D.; Aimone, J.B. Probabilistic neural computing with stochastic devices. *Adv. Mater.* **2022**, *35*, 2204569. [CrossRef] [PubMed]
26. Yeo, I.; Gi, S.-G.; Lee, B.-G.; Chu, M. Stochastic implementation of the activation function for artificial neural networks. In Proceedings of the 2016 IEEE Biomedical Circuits and Systems Conference (BioCAS), Shanghai, China, 17–19 October 2016; pp. 440–443.
27. Xie, Y.; Liao, S.; Yuan, B.; Wang, Y.; Wang, Z. Fully-parallel area- efficient deep neural network design using stochastic computing. *IEEE Trans. Circuits Syst. II Express Briefs* **2017**, *64*, 1382–1386. [CrossRef]
28. Abdellatef, H.; Khalil-Hani, M.; Shaikh-Husin, N.; Ayat, S.O. Stochastic Computing Correlation Utilization in Convolutional Neural Network Basic Functions. *Telkomnika* **2018**, *16*, 2835–2843. [CrossRef]
29. Hojabr, R.; Givaki, K.; Tayaranian, S.M.R.; Esfahanian, P.; Khonsari, A.; Rahmati, D.; Najafi, M.H. Skippynn: An embedded stochastic-computing accelerator for convolutional neural networks. In Proceedings of the 2019 56th ACM/IEEE Design Automation Conference (DAC), Las Vegas, NV, USA, 2–6 June 2019; pp. 1–6.
30. Ashok, P.; Bala Tripura Sundari, B. Computational Analysis of stochastic arithmetic computing and stochastic activation function. *J. Phys. Conf. Ser.* **2022**, *2325*, 012032. [CrossRef]
31. Nair, N.B.; Anita, J.P. Design of Multistage Counters Using Linear Feedback Shift Register. In *Inventive Communication and Computational Technologies*; Springer: Singapore, 2022; pp. 161–173.
32. Mahendra, P.; Ramesh, S.R. FPGA Implementation of High Performance Precise Signed and Unsigned Multiplier using Ternary 6-LUT Architecture. In Proceedings of the 2022 International Conference on Inventive Computation Technologies (ICICT), Lalitpur, Nepal, 20–22 July 2022; pp. 202–207.
33. Mousouliotis, P.G.; Petrou, L.P. Cnn-grinder: From algorithmic to high-level synthesis descriptions of cnns for low-end-low-cost fpga socs. *Microprocess. Microsyst.* **2020**, *73*, 102990. [CrossRef]
34. Gay, M.; Burchard, J.; Horácek, J.; Messeng Ekossono, A.S.; Schubert, T.; Becker, B.; Kreuzer, M.; Polian, I. Small scale AES toolbox: Algebraic and propositional formulas, circuit-implementations and fault equations. In Proceedings of the 6th Conference on Trustworthy Manufacturing and Utilization of Secure Devices (TRUDEVICE 2016), Barcelona, Spain, 14–16 November 2016.
35. Ashok, P.; Bala Tripura Sundari, B. Implementation of stochastic computing in activation functions using stochastic arithmetic components. In Proceedings of the 2023 IEEE 8th International Conference for Convergence in Technology (I2CT), Pune, India, 7–9 April 2023; pp. 1–5.