Accurate and Energy-Efficient Implementation of Non-Linear Adder in Parallel Stochastic Computing Using Sorting Network

Yawen Zhang[1], Runsheng Wang[1*], Yixuan Hu[1], Weikang Qian[2*], Yanzhi Wang[3], Yuan Wang[1*], Ru Huang[1]
[1]*Institute of Microelectronics, Peking University, Beijing, P.R. China*
[2]*UM-SJTU Joint Institute & MoE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University, Shanghai, P.R. China*
[3]*Department of Electrical and Computer Engineering, Northeastern University, Boston, USA*
[*]Email: r.wang@pku.edu.cn, qianwk@sjtu.edu.cn, wangyuan@pku.edu.cn

***Abstract*—Recently, due to the high fault tolerance and low hardware cost, stochastic computing (SC)-based neural network (NN) accelerators have been widely studied. One big challenge of it is the implementation of accumulation and activation function. The existing designs have problems of low accuracy and high energy consumption. In this paper, based on a special type of stochastic encoding, the parallel thermometer coding, we propose an accurate design for the combination of the accumulation and non-linear function, which is called a non-linear adder. Dedicated designs for the non-linear adders with the common activation functions such as hyperbolic tangent ($\tanh$), logistic (or sigmoid), and rectified linear unit (ReLU) are proposed using the bitonic sorting network and the selective interconnect. The experimental results show that, at the cost of the area, the proposed non-linear adder achieves more than three orders of magnitude improvement in accuracy and at least $44.5\times$ energy consumption reduction compared with the traditional designs.**

I. Introduction

Stochastic computing (SC) [1], [2] has been explored as an alternative method to implement arithmetic calculations. It uses the probability in a bitstream to represent a value. It not only has lower power consumption and higher fault tolerance than conventional binary radix computing, but also enables a higher integration density [3]. Therefore, SC has been applied to many neural network (NN) computations such as multi-layer perceptrons [4], [5], deep belief networks [6], and deep neural networks [7]–[12], where a large number of multiplication operations make SC very attractive.

However, one big challenge for the SC-based NN accelerators is to accurately and efficiently realize the accumulation and activation function. In what follows, we call the combination of the accumulation and the activation function the *non-linear addition*. So far, the SC-based implementation of the accumulation uses either an approximate parallel counter (APC) or a MUX-based scaled adder [13], while that of the activation functions uses a counter-based design [14]. However, they are both imprecise. As a result, the accuracy of an SC-based NN accelerator can easily be reduced to an unacceptable level. In order to improve the accuracy, a bitstream long enough is usually used. However, this causes a long latency and a large energy consumption for the whole NN accelerator, weakening the advantage of SC. Furthermore, to the best of the authors' knowledge, there is no solution to implement the exact rectified linear unit (ReLU) function unless the bitstream is converted into the binary format. However, an exact ReLU is an essential part of the state-of-the-art NNs [15].

In this paper, we propose a novel non-linear adder working with a special type of stochastic encoding, the parallel thermometer coding. It is based on the bitonic sorting network and the selective interconnect. Bitonic sorting network [16], [17] transforms multiple parallel bitstreams into one. By selecting different outputs of the sorting network, three widely-used activation functions, hyperbolic tangent ($\tanh$), logistic (or sigmoid), and ReLU functions, are accurately implemented. We summarize our contributions as follows:

- We propose a novel non-linear adder based on parallel thermometer coding. The adder is implemented by a bitonic sorting network and a selective interconnect. It significantly improves the latency and the accuracy.
- Compared with the traditional SC implementations of the non-linear adder, the proposed one reduces the energy consumption by at least $44.5\times$.

The rest of the paper is organized as follows. Section II discusses the related works. Section III introduces the architecture of the proposed non-linear adder. Section IV presents the experimental results. Finally, Section V concludes the paper.

II. Related Works

There are two designs of the non-linear adder in SC. They differ in the implementations of the adder. One is based on APC, and the other is based on MUX. In the following, we will give a brief introduction of the two designs.

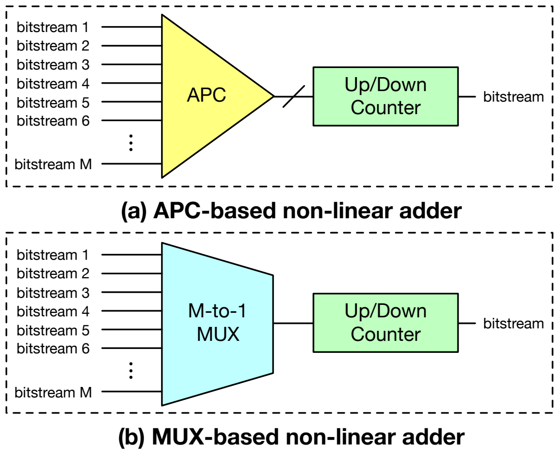

Fig. 1. Two traditional non-linear adders in SC.

A. APC-Based Non-Linear Adder

Fig. 1(a) illustrates a non-linear adder based on APC. Assume that the number of input bitstreams is M. The APC calculates the sum of the M input bits in each clock cycle. Since the weight and input data in a neural network can be either positive or negative, the bipolar stochastic encoding is used. The output sum of the APC in the binary format is sent to an up/down saturation counter to realize the non-linear activation function. By different configurations of the counter, various activation functions can be realized [14].

B. MUX-Based Non-Linear Adder

Fig. 1(b) shows a MUX-based non-linear adder, which consists of an M-to-1 MUX and an up/down saturation counter. The MUX realizes an addition of the M input bitstreams with a scaling factor of $1/M$. The output of the MUX is fed into a counter, and different activation functions are realized by different configurations of the counter. Since the weights and the inputs of a neural network can be either positive or negative, the bitstream adopts the bipolar format.

Unfortunately, the two existing schemes suffer from serious problems. For the MUX-based scaled adder, its accuracy is far less than that of the APC-based adder, due to the randomness in the selecting signal of the MUX [13]. Although APC can achieve a more accurate addition, it needs an additional module to convert the bitstreams into a binary number, which eliminates the advantages of SC including simple circuitry and high fault tolerance. In addition, the accuracy of the non-linear activation function implemented by the counter highly depends on the randomness of the input bitstream. An input bitstream not random enough reduces the computation accuracy.

III. Proposed Architecture

In this section, we describe the architecture of the proposed non-linear adder based on thermometer coding. It consists of two parts, as shown in Fig. 2. The first part is a bitonic sorting network, which converts M parallel input bitstreams with the bitstream length (BSL) of N into a longer bitstream with the BSL of MN. The second part is a selective interconnect, which generates different non-linear addition results by selecting different outputs of the sorting network.

A. Thermometer Coding

Thermometer coding is a type of unary coding. It consists of a continuous sequence of 1s followed by a continuous sequence of 0s. For example, 11111000 is a bitstream with thermometer coding. Since the input data in a neural network can be either positive or negative, the bipolar-format thermometer coding is used in this work. In the bipolar format, a bitstream with the ratio of 1s as x represents the value $(2x-1)$. For example, the bitstreams 11111000 and 11100000 encode the values $1/4$ and $-1/4$, respectively. Furthermore, we use the parallel thermometer coding [18]–[21], where all the bits of a stream are simultaneously input to and output from a circuit. The computation finishes in one clock cycle.

B. Bitonic Sorting Network

Bitonic sorting network is a parallel implementation of sorting, as shown in Fig. 2. It is composed of many arrow modules, each comparing two inputs. There are two types of arrow modules giving different output orders. They are ascending sorting module and descending sorting module. In traditional binary computing, each arrow module consists of a multi-bit comparator and two multi-bit multiplexers. However, in SC, since each arrow module compares two single-bit inputs, its design is very simple and only consists of an AND gate and an OR gate, as shown in Fig. 2. In a bitonic sorting network, an unordered sequence of length L is first transformed into an ascending and a descending sequence of length $L/2$ by merge sorting, and then these two sequences are transformed into a monotonic sequence.

The parallel bitstreams are input into a bitonic sorting network simultaneously. After one clock cycle, the sorting result of all bits of the input streams is obtained. As shown in Fig. 2, four input bitstreams with the BSL of 4 are transformed into a longer bitstream with the BSL of 4×4 through the sorting network. It is worth noting that the output bitstream of the sorting network is arranged in a descending order from the top to the bottom. It is a longer bitstream based on thermometer coding and can be seen as a scaled addition of the input bitstreams with a scaling factor of $1/M$, where M in the number of input bitstreams. For example, in Fig. 2, if the four input bitstreams are 0000, 1000, 1110, and 0000, which encode -1, -0.5, 0.5, and -1, respectively, then the output is a sorted result 1111000000000000, which encodes

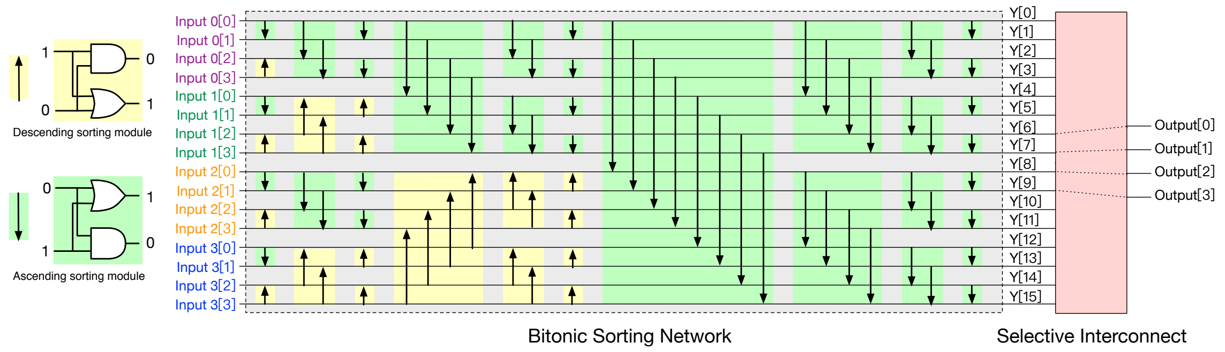

Fig. 2. The architecture of the proposed non-linear adder, which consists of a bitonic sorting network and a selective interconnect.

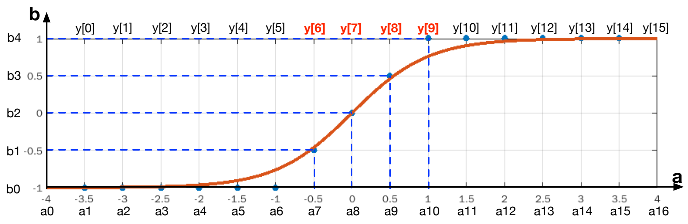

Fig. 3. An example for illustrating our selective interconnect method. The example is the tanh function with the BSL of 4 and the number of input bitstreams of 4.

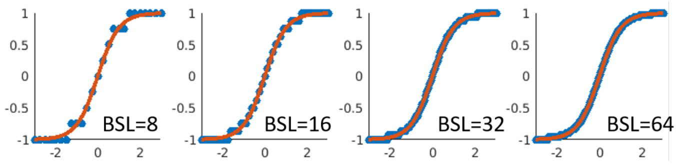

Fig. 4. The accuracy of the proposed non-linear adder with the tanh activation function for different bitstream lengths (BSLs) from 8 to 64. The red lines are the expected tanh curve and the blue dots are the outputs of the proposed design.

-0.5. It can be seen that the output is the sum of the input values scaled by $1/4$. The actual sum can be easily obtained by multiplying the scaled sum by M.

C. Selective Interconnect

By observing the output results of the sorting network, we can see that the output of each bit is a comparison result. For M input bitstreams with the BSL of N, if the output $y[i]$ of the sorting network is 1, it means that the output bitstream has more than i 1s out of MN bits. This indicates that the scaled sum of the inputs is larger than $\frac{2i}{MN} - 1$, or the actual sum is larger than $\frac{2i}{N} - M$; otherwise, it means that the actual sum is smaller than or equal to $\frac{2i}{N} - M$. Based on this observation and inspired by a method described in [22], we propose a selective interconnect method, which chooses N outputs from the MN outputs $y[i]$ of the sorting network as the final outputs to implement the non-linear activation function. Notice that the method in [22] implements a function with thermometer coding and the BSLs of its input and output are the same. However, our method handles the situation where the BSLs of the input and the output are different.

We use the tanh function as an example to illustrate our method. We assume that the number of the input bitstreams is $M = 4$ and the BSL is $N = 4$. The horizontal axis a represents the sum of the inputs, ranging from -4 to 4. It is the input to the tanh function. We divide the range $[-4, 4]$ into $MN + 1 = 17$ equidistant points $(a_0, a_1, \ldots, a_{16}) = (-4, -3.5, \ldots, 4)$. Except the first point a_0, we link each remaining point a_{i+1} $(0 \leq i \leq 15)$ to the output $y[i]$ of the sorting network, as shown in Fig. 3. By the previous observation, we can see that if the input to the tanh function changes from a_i (i.e., $\frac{i}{2} - 4$) to a_{i+1} (i.e., $\frac{i+1}{2} - 4$), then the output $y[i]$ flips from 0 to 1. The vertical axis b represents the output of the tanh function, ranging from -1 to 1. We divide the range $[-1, 1]$ into $N + 1 = 5$ equidistant points $(b_0, b_1, \ldots, b_4) = (-1, -0.5, \ldots, 1)$. To decide which $y[i]$'s should be selected as the final outputs, we further discretize the tanh curve. For each a_i $(0 \leq i \leq 16)$, we round $\tanh(a_i)$ to the closest point $f(a_i) \in \{b_0, b_1, \ldots, b_4\}$. This gives the 17 blue dots $(a_0, f(a_0)), \ldots, (a_{16}, f(a_{16}))$ in Fig. 3. Then, we scan these dots from left to right. If $f(a_{i+1})$ increases over $f(a_i)$ by a value of $2k/N \geq 2/N$, then it means that with the input value changing from a_i to a_{i+1}, the output value increases by $2k/N$. Note that the input value changing from a_i to a_{i+1} is indicated by the output $y[i]$ flipping from 0 to 1. Furthermore, since the output is also in bipolar thermometer coding, the increase of the output value by $2k/N$ indicates that the number of 1s in the output bitstream should increase by k. Given these two facts, we simply choose k $y[i]$'s as the outputs. In our example, since $f(a_7)$ increases over $f(a_6)$ by a value of $(2 \cdot 1)/4$, $y[6]$ is selected as a final output. The final 4 outputs are $y[6], y[7], y[8], y[9]$ for this example, as highlighted in red in Fig. 3.

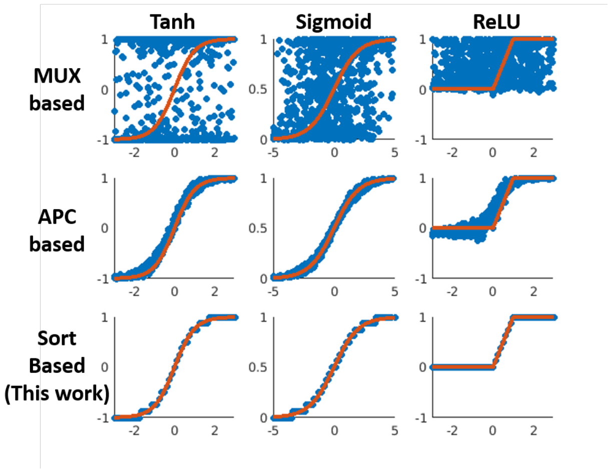

Fig. 5. The accuracy of three non-linear adders with 16 input bitstreams. The red lines are the expected curves and the blue dots are the outputs of the non-linear adders.

Fig. 4 shows the implementation result of the tanh activation function with different BSLs. As the BSL increases, the accuracy of the proposed non-linear adder improves. It is worth noting that for the three common activation functions, only the middle part of the sorting network outputs is selected as the final outputs. Therefore, some arrow modules in the later layers of the sorting network can be omitted since they do not contribute to the selected outputs, thus reducing the area.

IV. Experimental Results

A. Accuracy

Compared with the traditional SC-based non-linear adders, our proposed design is accurate and deterministic. Its accuracy loss is only due to the insufficient BSL. Fig. 5 shows the results of three non-linear adders with three different activation functions. The number of input bitstreams is 16. For the traditional implementations, the BSL is 1024, while for our design, the BSL is 16.

For the MUX-based non-linear addition, in each clock cycle, one of the M input bitstreams is randomly selected as the output, while the information carried by the other $M - 1$

TABLE I
COMPARISON OF DIFFERENT NON-LINEAR ADDERS.

	Non-Linear Function	Variance (%)	Area (μm^2)	Power (μW)	Latency/Operation (μs)	Energy/Operation (fJ)
MUX-based (1024 BSL)	Tanh	105.45	135.65	18.7	10.24	191.5
	Sigmoid	17.65	131.41	18.9	10.24	193.5
	ReLU	22.51	115.01	11.3	10.24	115.7
APC-based (1024 BSL)	Tanh	0.35	261.25	25.4	10.24	260.1
	Sigmoid	0.5	106.37	18.7	10.24	191.5
	ReLU	1.78	253.31	22.2	10.24	227.3
This work (16 BSL)	Tanh	0.08	5607.58	701	0.01	7.0
	Sigmoid	0.04	5602.11	609	0.01	6.1
	ReLU	0	5354.62	693	0.01	6.9
This work (8 BSL)	Tanh	0.29	2082.93	250	0.01	2.5
	Sigmoid	0.13	2009.73	210	0.01	2.1
	ReLU	0	1981.15	258	0.01	2.6

bitstreams is lost. With the increase in the number of input bitstreams, the error caused by the information loss becomes larger. As shown in Fig. 5, for 16 input bitstreams, the MUX-based non-linear addition cannot guarantee accurate output results. For the APC-based non-linear addition, although the stochastic-to-binary conversion improves the accuracy of the addition operation, the counter-based implementation of the non-linear activation function needs a large BSL to achieve relatively accurate results. Even with a BSL of 1024, the non-linear function still randomly fluctuates. Especially for the ReLU function, there are large errors for x near zero.

Compared with the existing SC implementations, the proposed design is accurate and only has rounding errors. With only a small BSL, such as 16, it can achieve accurate non-linear functions, especially for the piecewise linear function like ReLU.

B. *Overall Performance*

TABLE I compares the proposed non-linear adder with the traditional MUX-based and APC-based non-linear adders, in terms of variance, area, power, latency per operation, and energy per operation for three commonly-used activation functions. We synthesize different non-linear adders using Synopsys Design Compiler with TSMC 40nm technology. The operating frequency is 100MHz.

The results indicate that, compared with the two traditional designs, the proposed one reduces the variance by more than three orders of magnitude and improves the accuracy of three kinds of non-linear addition. Our design with the BSL of 16 is more accurate than the traditional designs with the BSL of 1024. Even for a particularly small BSL of 8, our design is still more accurate than the traditional ones. Especially for ReLU, a piecewise linear function, our design can achieve accurate results, which means that the variance is 0, without considering the quantization error due to the limited BSL.

The proposed parallel design significantly reduces the latency from BSL clock cycles to only one cycle. Nevertheless, the latency is a trade-off with the area. As shown in TABLE I, even for the BSL of 8, the area of the proposed design is at least $7.8\times$ larger than that of the APC-based design and $15.3\times$ larger than that of the MUX-based design. This increase in area is because our design method expands the input parallelism by a factor of the BSL. Moreover, the increase in area results in a substantial increase in static power consumption. The proposed design with the BSL of 8 has at least $9.8\times$ and $11.1\times$ more power consumption than the APC-based and MUX-based designs, respectively.

Given the reduction of latency and the increase of power consumption, in order to make a fair comparison, we further compare the overall energy consumption. The proposed adder with the BSL of 8 achieves at least $44.5\times$ and $87.4\times$ energy improvement over the MUX-based one and the APC-based one, respectively. The reason is because we adopt parallel thermometer coding, which has no intrinsic errors (e.g., random fluctuation error) except the rounding error, and thus improves the computation accuracy. Therefore, a small BSL can meet the accuracy requirement of arithmetic calculations. Such a decrease in BSL reduces the energy consumption significantly.

In summary, the proposed non-linear adder achieves high accuracy. Although it has a larger area, the total energy consumption is reduced by more than an order of magnitude compared with the traditional designs.

V. CONCLUSION

In this paper, we propose a new non-linear adder based on parallel stochastic computing, where all calculations are implemented by a bitonic sorting network and a selective interconnect. Compared with the traditional designs, ours is accurate and deterministic with only rounding errors. For a short BSL, such as 8 or 16, the accuracy improves by more than three orders of magnitude compared with the traditional designs. Particularly, the ReLU function has no error. The experimental results show that the proposed non-linear adder achieves at least $44.5\times$ and $87.4\times$ energy consumption improvement compared with the MUX-based one and the APC-based one, respectively.

ACKNOWLEDGMENT

This work is supported by the National Key Research and Development Program of China (Grant No. 2018YFE0203801), NSFC (61522402 and 61421005) and the 111 Project (B18001).

References

[1] B. R. Gaines, “Stochastic computing systems.” Advances in information systems science. Springer, Boston, MA, 1969, pp. 37-172.

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

[3] Y. Zhang et al., “Design guidelines of stochastic computing based on FinFET: A technology-circuit perspective,” IEEE International Electron Devices Meeting (IEDM), 2017, pp. 6.6.1-6.6.4.

[4] V. T. Lee, A. Alaghi, J. P. Hayes, V. Sathe and L. Ceze, “Energy-efficient hybrid stochastic-binary neural networks for near-sensor computing,” Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017, pp. 13-18.

[5] J. L. Rosselló, V. Canals, A. Morro, “Probabilistic-based neural network implementation,” IEEE International Joint Conference on Neural Networks (IJCNN), 2012, pp. 1-7.

[6] K. Sanni, G. Garreau, J. L. Molin and A. G. Andreou, “FPGA implementation of a deep belief network architecture for character recognition using stochastic computation,” 49th Annual Conference on Information Sciences and Systems (CISS), 2015, pp. 1-5.

[7] Z. Li, A. Ren, J. Li, Q. Qiu, Y. Wang and B. Yuan, “DSCNN: Hardware-oriented optimization for stochastic computing based deep convolutional neural networks,” IEEE 34th International Conference on Computer Design (ICCD), 2016, pp. 678-681.

[8] K. Kim, J. Kim, J. Yu, J. Seo, J. Lee and K. Choi, “Dynamic energy-accuracy trade-off using stochastic computing in deep neural networks,” 53nd ACM/EDAC/IEEE Design Automation Conference (DAC), 2016, pp. 1-6.

[9] A. Ren, Z. Li, C. Ding, Q. Qiu, Y. Wang, J. Li, X. Qian and B. Yuan, “SC-DCNN: Highly-scalable deep convolutional neural network using stochastic computing,” ACM SIGOPS Operating Systems Review, vol. 25, no. 5, pp. 405-418, 2017.

[10] A. Ardakani, F. Leduc-Primeau, N. Onizawa, T. Hanyu and W. J. Gross, “VLSI implementation of deep neural network using integral stochastic computing,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 10, pp. 2688-2699, Oct. 2017.

[11] Y. Zhang et al., “When sorting network meets parallel bitstreams: A fault-tolerant parallel ternary neural network (TNN) accelerator based on stochastic computing,” Design, Automation & Test in Europe Conference & Exhibition (DATE), 2020.

[12] Y. Zhang et al., “Parallel convolutional neural network (CNN) accelerators based on stochastic computing,” IEEE International Workshop on Signal Processing Systems (SiPS), 2019.

[13] J. Li et al., ”Towards acceleration of deep convolutional neural networks using stochastic computing,” 22nd Asia and South Pacific Design Automation Conference (ASP-DAC), 2017, pp. 115-120.

[14] J. Li et al., “Hardware-driven nonlinear activation for stochastic computing based deep convolutional neural networks,” International Joint Conference on Neural Networks (IJCNN), 2017, pp. 1230-1236.

[15] K. He, X. Zhang, S. Ren and J. Sun, ”Delving deep into rectifiers: surpassing human-level performance on imageNet classification,” IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1026-1034.

[16] M. H. Najafi, D. J. Lilja, M. D. Riedel and K. Bazargan, “Low-cost sorting network circuits using unary processing,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 8, pp. 1471-1480, Aug. 2018.

[17] M. H. Najafi, D. J. Lilja, M. Riedel and K. Bazargan, “Power and area efficient sorting networks using unary processing,” IEEE International Conference on Computer Design (ICCD), 2017, pp. 125-128.

[18] D. Jenson and M. Riedel, “A deterministic approach to stochastic computation,” IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2016, pp. 1-8.

[19] Y. Zhang et al., “A parallel bitstream generator for stochastic computing,” Silicon Nanoelectronics Workshop (SNW), 2019, pp. 1-2.

[20] W. Qian, R. Wang, Y. Wang, M. Riedel and R. Huang, “A survey of computation-driven data encoding,” IEEE International Workshop on Signal Processing Systems (SiPS), 2019.

[21] X. Zhang et al., ”Memory system designed for multiply-accumulate (MAC) engine based on stochastic computing,” International Conference on IC Design and Technology (ICICDT), 2019, pp. 1-4.

[22] S. Mohajer, Z. Wang and K. Bazargan, “Routing magic: Performing computations using routing networks and voting logic on unary encoded data,” ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA), 2018, pp. 77-86.