International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 9 Sep 2021, pp: 938-942 www.ijaem.net ISSN: 2395-5252

Improved Logarithmic Multiplier Using Verilog

Sayyedhussain Siddiqui and Dr.Kiran V

Submitted: 05-09-2021 Revised: 12-09-2021 Accepted: 15-09-2021

ABSTRACT—Multiplication is the most resource-hungry operation in neural networks (NNs). Logarithmic multipliers (LMs) simplify multiplication to shift and addition operations and thus reduce the energy consumption. Since implementing the logarithm in a compact circuit often introduces approximation, some accuracy loss is inevitable in LMs. However, this loss is tolerated by the inherent error tolerance of NNs and their associated applications. This article proposes an improved logarithmic multiplier (ILM) that, unlike existing designs, rounds both inputs to their nearest powers of two by using a proposed nearest-one detector (NOD) circuit. Considering that the output of the NOD uses a one-hot representation, some entries in the truth table of a conventional adder cannot occur. Hence, a compact adder is designed for the reduced truth table.

I. INTRODUCTION

Artificial neuron

Neurons are the main processing units of NNs that compute a weighted sum of their inputs and send the result through an activation function (AF). The AF introduces non-linearity into a neuron's behaviour and maps the resulting output values either into either the interval (-1, 1) or (0, 1) [10]. The AF can be either a hard-limiting (e.g., a step function) or a soft-limiting function (e.g., a sigmoid function) [8]. Fig. 1 shows the structure of an artificial neuron. A neuron has $n \geq 2$ inputs (depending on the network structure) and one output. Each input $x_i$is multiplied by its corresponding synaptic weight w_i; i = 0; 1; :::; n. An adder tree is then used to sum up the products. The resulting sum is then input to the AF. An external bias b is often added to increase or lower the input value of the AF [20].

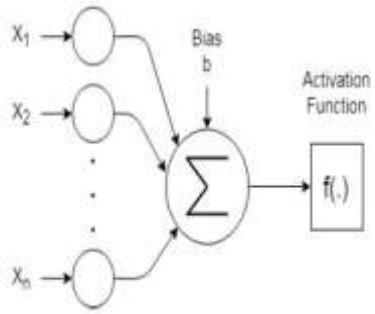


Fig. 1: Model of an artificial neuron.

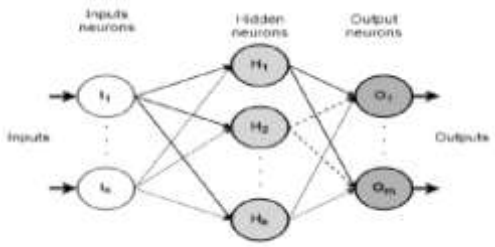


Fig. 2: Structure of a feed-forward NN.

Feed-forward neural networks

The two major operating modes for NNs are training and inference. The training process is usually performed infrequently and off-line and, therefore, its energy consumption is less of a concern [8]. The inference process, on the other hand, is done frequently. Although it is less computation intensive than the training process, inference still requires significant computation for large networks. Fig. 2 shows a simple feed-forward NN with n, k, and m neurons in the input, hidden, and output layers,respectively.

Synaptic weights in neuralnetworks

The n multiplications of $x_i \times w_i$in Fig. 1, where $1 \leq i \leq n$, are the main bottleneck in the performance of NNs. Hence, it is helpful to have insight into the synaptic weights when designing multipliers for NNs. Here are the three main observations that have been found to be helpful:

- Past research has investigated the effects of

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 9 Sep 2021, pp: 925-930 www.ijaem.net ISSN: 2395-5252

reduced precision on the performance of NNs and has found that the precision can be safely reduced without significant negative impact on the accuracy of NNs [5], [6],[7].

- The authors in [21] showed that the trained weights in NNs are mostly centered near zero. Hence, even after up-scaling, the chances of having large weights, i.e. weights at both the positive and negative ends of the scaled spectrum, aresmall.
- Large synaptic weights are not desirable in NNs and it is common to use weight decay techniques to reduce them. During weight decay, after each update, the weights are multiplied by a factor slightly less than 1 to prevent them from growing too large[22].

Consequently, large weights are unlikely to appear in trained NNs and limiting them should not significantly influence the performance of NNs. We exploit this insight in the design of the proposed multiplier, more specifically in the design of the NOD circuit.

Improved Logarithmic Multipliers:

Here we propose a method to approximate $\log_2 N$ which, unlike the existing approaches, has a double-sided error distribution and can be used as a more accurate baseline design instead of the Mitchell approach. The existing techniques in the literature for improving the accuracy of the Mitchell method are also applicable to the proposed method.

Any positive integer N can be also represented as:
$N = 2^{k+1}(1 - y)$ where $0 < y \leq 1$.
The conventional logarithmic approximation uses the highest power of two smaller than the given numberN.
Instead, we propose the approximation given in Algorithm

1. Note that $2^k \leq N < 2^{k+1}$. As shown in Algorithm 1, when $N - 2^k < 2^{(k+1)} - N$ we underestimate the value of $\log_2 N$ as k; otherwise,
we overestimate it as k +1.

The exact, Mitchell, and the proposed methods for computing $\log_2 N$ are plotted in Fig. 3. The k parameter values corresponding to the N values, obtained from Algorithm 1, are also shown in Fig. 3. Only $k \geq 3$ is shown in order to keep the figure clear and easy to read. One power-of-two interval, i.e. k = 6, is also enlarged and depicted as an inset in Fig. 3 to better show the behaviourof the two approximate methods compared to the exact $\log_2 N$ function.

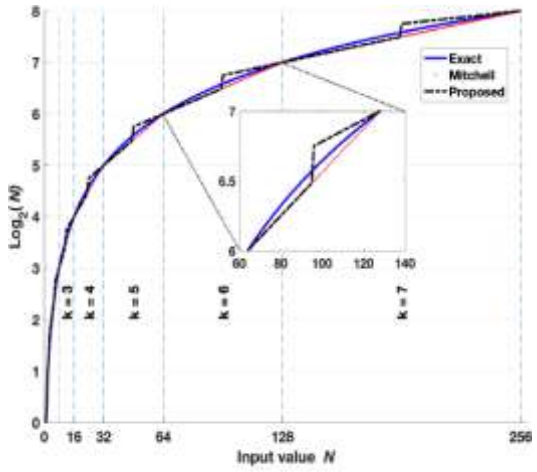


Fig. 3: Approximation of $log_2 N$.

Note that the proposed approximation results in a more than 6 × smaller average error (over the range [1, 255]) than the Mitchell method (0.0088 vs. 0.0568), which is due to the double-sided error distribution of the proposed approach. With respect to the root mean square error (RMSE), the Mitchell approach is slightly more accurate than the proposed method (0.0627 vs. 0.0794). It is also evident in Fig. 3 that the magnitude of the error can be larger than the Mitchell method for the proposed method. However, the error analysis of the resulting multipliers with respect to two other error metrics, i.e. the common mean relative error distance (MRED) and the normalized mean error distance (NMED, by the maximum output of the accurate design) ([24], [25], [26]) shows that the proposed designs are actually more accurate. This occurs because using the proposed method for approximating $\log_2 N$, the errors are likely to cancel out each other and, therefore, the proposed multiplier's accuracy increases.

II. METHODOLOGY:

High-level description of the ILM design

The proposed ILM first transforms the multiplicand A and multiplier B to the closest powers of two plus an additional term, as givenby:

As shown in (3), the three most significant terms are all multiples of powers of two that can be efficiently implemented as left-shift operations in hardware. In this design, the least significant term (q1q2) is ignored and left as the approximation error. A more detailed description of the ILM is

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 9 Sep 2021, pp: 925-930 www.ijaem.net ISSN: 2395-5252

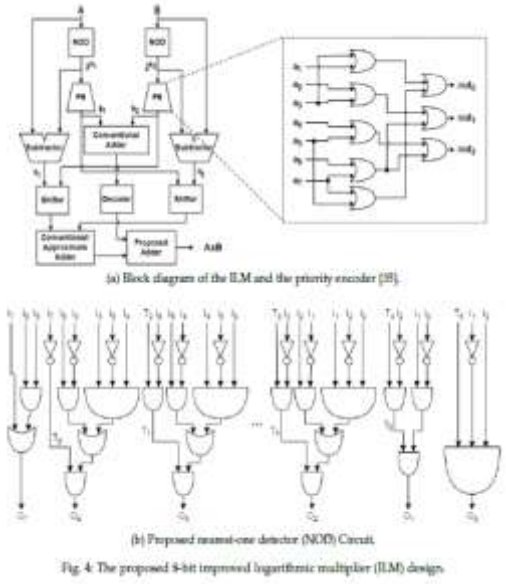
provided in Algorithm 2, where NOD, PE and DEC denote the nearest-one detector, the priority encoder, and the decoder, respectively. Detailed descriptions of these three components are given in the following subsection.

Hardware Implementation

The ILM can be implemented by either: (1) implementing the logic to calculate the nearest powers of two, or (2) using look-up tables (LUTs). We decided not to use LUTs as that would increase the memory usage, which is often a serious bottleneck for neural network applications [4], [8]. The block diagram of the 8-bit ILM is given in Fig. 4(a). The NOD circuits (Figs. 4(b) and 4(c)) are based on a leading-one detector (LOD) circuit. However, unlike the LOD, the NODs find the nearest power of two to a given input. Similar to some existing LODs [27], [28], the proposed NODs evaluate from the MSB to the LSB. The priority encoder (PE) in Fig. 4(a) determines the number of required shifts based on the NOD's output. The two residue terms q1 and q2 are also calculated and shifted according to the k2 and k1 values, respectively, and a decoder generates the most significant term, 2^{k1+k2} . Finally, the three resulting terms are summed up to obtain the approximate product. For hardware savings, we used the PE proposed in [23]. Fig. 4(b) depicts the design of the proposed NOD, where I and O are the primary input and output signals, respectively. The design is a simplified version of the NOD in [19]. Normally, nine bits are needed to represent the nearest power of two to an 8-bitinput.

However, as previously discussed, large synaptic weights are unlikely to appear in trained NNs and removing them would not significantly influence the performance of a NN. Hence, the NOD in [19] is simplified by rounding down the output of the NOD to the largest power of two representable in 8 bits, i.e. 128. In other words, up-rounding is not performed if the nearest power of two is greater than 128. This will not have a significant detrimental impact on the performance ofNNs.

Note that the ILM needs to use the sign-magnitude representation for applications that require signed multiplications. It may not be as hardware-efficient as the 2's complement representation when multiply-accumulate operations are required. However, it is widely-used for logarithmic and non-logarithmicarithmetic circuits that are designed for unsigned numbers, e.g. [8], [9], [10]. Using this method, the sign bits of the two input operands are XOR-ed to obtain the sign of the final product and only the magnitude is computed using the described designhere.



(a) Block diagram of the ILM and the priority encoder [19].

(b) Proposed nearest-one detector (NOD) Circuit.

Fig. 4: The proposed 8-bit improved logarithmic multiplier (ILM) design.

In order to further improve the hardware efficiency, we also propose a novel adder. This adder is used in the final stage, i.e. the adder that produces A × B in Fig. 4(a). There are three inputs to this adder (i.e., $2^{k1+k2,}$ $q1\times2^{k2}$, and $q2\times2^{k1}$, step 12 in Algorithm 2), hence an adder tree composed of two adders is required. The conventional 8- bit adder (composed of conventional FAs) is used to add $q1\times2^{k2}$ and $q2\times2^{k1}$ and the proposed adder is used to add the result to the third term, 2^{k1+k2}

Fig. 4(a). Note that the proposed adder is not an approximate design, however it has a simplified structure. Since 2^{k1+k2} is in a one-hot representation, the structure of the 8-bit adder can be modified and simplified accordingly. The truth tables for both the conventional and the proposed FAs are shown in Table 1. Note that the "not applicable" (N/A) entries in Table 1 cannot happen because there is only one '1' in one of the inputs. If A is a one hot number and the '1' is at bit position i, then it is not possible to have a carry in from less significantpositions.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 9 Sep 2021, pp: 925-930 www.ijaem.net ISSN: 2395-5252

TABLE 1: The proposed vs. conventional full adder.

a	b	c_{in}	Conventional FA		Proposed FA	
			sum	c_{out}	sum	c_{out}
0	0	0	0	0	0	0
0	0	1	1	0	1	0
0	1	0	1	0	1	0
0	1	1	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	1	N/A	N/A
1	1	0	0	1	0	1
1	1	1	1	1	N/A	N/A
Conventional	$sum = (\bar{a}.\bar{b}.c_{in}) + (\bar{a}.b.\overline{c_{in}}) + (a.\bar{b}.\overline{c_{in}}) + (a.b.c_{in})$ $c_{out} = (a.b) + (a.c_{in}) + (b.c_{in})$					
Proposed	$sum = (\bar{b}.c_{in}) + (\bar{a}.b.\overline{c_{in}}) + (a.\bar{b})$ $c_{out} = (a.b) + (b.c_{in})$					

III. RESULTS:

Fig. 5: Simulation Results of ILM

As shown in the Fig. 5, the results of the ILM match with the proposed algorithm and thus the hardware implementation is done and validated. Fig. 6 shows the synthesized circuit for this algorithm.

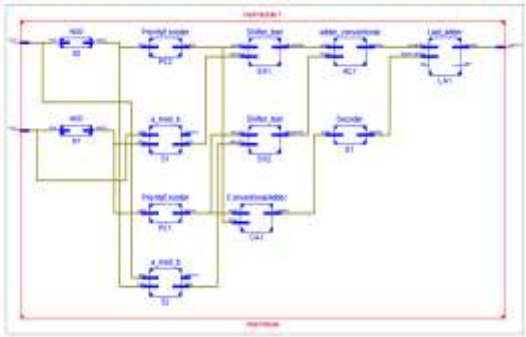


Fig. 6: Synthesized Circuit of proposed ILM

The proposed adder is also integrated in the design and is named as "last adder".

IV. CONCLUSION:

This work proposes a novel approximation method to ef- ficiently compute $\log_2 N$. Using this method, an improved logarithmic multipliers (ILM) is designed. The proposed ILM is more accurate and has the smallest MRED values compared to other logarithmic designs in theliterature. We observed that a few LSBs can be approximated in the ILM for saving hardware without a significant effect on its accuracy. For example, ILM with five approximation bits, ILM-5 can be 6.78%-17.48% more power-efficient and up to 6.07% smaller than the recent design in [17]. Finally, two well-known NNs were considered as benchmark applica- tions, for which the proposed designs show a higher top-1 classification accuracy than the other designs. The exact multipliers in both NNs were replaced with LMs and the ILM-5 resulted in the most energy-efficient NN structure. Interestingly, higher classification accuracies are obtained for the CIFAR-10 dataset by using the ILM compared to the use of exact (and other LM)multipliers

REFERENCE:

[1] J. Schmidhuber, "Deep learning in neural networks: An overview,"Neural Networks, vol. 61, pp. 85–117,2015.

[2] L. Shao, D. Wu, and X. Li, "Learning deep and wide: A spectral method for learning deep networks," IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 12, pp. 2303–2308,2014.

[3] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations." Journal of Machine Learning Research, vol. 18, pp. 187–1, 2017.

[4] G. Srinivasan, P. Wijesinghe, S. S. Sarwar, A. Jaiswal, and K. Roy, "Significance driven hybrid 8T-6T SRAM for energy-efficient synaptic storage in artificial neural networks," Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 151–156,2016.

[5] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers et al., "In-datacenter performance analysis of a tensor processing unit," ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA), pp. 1–12,2017.

[6] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," International Conference on Machine Learning, pp. 1737–1746,2015.

[7] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," arXiv preprint arXiv:1609.08144,2016.

[8] S. S. Sarwar, S. Venkataramani, A. Ankit,

A. Raghunathan, and K. Roy, "Energy-efficient neural computing with approximate multipliers," ACM Journal on Emerging Technologies in Computing Systems (JETC), vol. 14, no. 2, p. 16,2018.

[9] U. Lotriˇc and P. Buli´c, "Applicability of approximate multipliers in hardware neural networks," Neurocomputing, vol. 96, pp. 57–65,2012.

[10] V. Mrazek, S. S. Sarwar, L. Sekanina, Z. Vasicek, and K. Roy, "Design of power-efficient approximate multipliers for approximate artificial neural networks," IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1–7,2016.

[11] V. Paliouras and T. Stouraitis, "Low-power properties of the logarithmic number system," 15th IEEE Symposium on Computer Arithmetic, pp. 229–236,2001.

[12] S. Gandhi, M. S. Ansari, B. F. Cockburn, and J. Han, "Approximate leading one detector design for a hardware- efficient Mitchell multiplier," IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), pp. 1– 4, 2019

[13] J. Y. L. Low and C. C. Jong, "Unified Mitchell-based approximation for efficient logarithmic conversion circuit," IEEE Transactions on Computers, vol. 64, no. 6, pp. 1783–1797,2015

[14] W. Liu, J. Xu, D. Wang, C. Wang, P. Montuschi, and F. Lombardi, "Design and evaluation of approximate logarithmic multipliers for low power error-tolerant applications," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 9, pp. 2856–2868,2018.

[15] D. De Caro, N. Petra, and A. G. Strollo, "Efficient logarithmic converters for digital signal processing applications," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 58, no. 10, pp. 667–671,2011.

[16] T. B. Juang, S.-H. Chen, and H.-J. Cheng, "A lower error and ROM free logarithmic converter for digital signal processing applications," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 56, no. 12, pp. 931– 935,2009.

[17] B.-G. Nam, H. Kim, and H.-J. Yoo, "Power and area-efficient unified computation of vector and elementary functions for handheld 3D graphics systems," IEEE Transactions on Computers, vol. 57, pp. 490–504,2008.

[18] J. N. Mitchell, "Computer multiplication and division using binary logarithms," IRE Transactions on Electronic Computers, no. 4, pp. 512–517,1962.

[19] M. S. Ansari, B. Cockburn, and J. Han, "A hardware-efficient logarithmic multiplier with improved accuracy," Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 922–925,2019.

[20] B. C. Cs´aji, "Approximation with artificial neural networks," Faculty of Sciences, EtvsLornd University, Hungary, vol. 24, p. 48,2001.

[21] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," arXiv preprint arXiv: 1505.05424,2015.

[22] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing coadaptation of feature detectors," arXiv preprint arXiv: 1207.0580,2012.

[23] M. S. Kim, A. A. Del Barrio, R. Hermida, and N. Bagherzadeh, "Low-power implementation of Mitchell's approximate logarithmic multiplication for convolutional neural networks," 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 617– 622,2018.

[24] H. Jiang, C. Liu, L. Liu, F. Lombardi, and J. Han, "A review, classification, and comparative evaluation of approximate arithmetic circuits," ACM Journal on Emerging Technologies in Computing Systems (JETC), vol. 13, no. 4, p. 60,2017

[25] H. Jiang, F. J. Santiago, M. S. Ansari, L. Liu, B. F. Cockburn, F. Lombardi, and J. Han, "Characterizing approximate adders and multipliers optimized under different design constraints," pp. 393–398,2019.

[26] M. S. Ansari, H. Jiang, B. F. Cockburn, and J. Han, "Low-power approximate multipliers using encoded partial products and approximate compressors," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 8, no. 3, pp. 404–416,2018.

[27] K. H. Abed and R. E. Siferd, "CMOS VLSI implementation of a low-power logarithmic converter," IEEE Transactions on Computers, vol. 52, no. 11, pp. 1421–1433,2003.