IPSJ Transactions on System LSI Design Methodology Vol.15 16–19 (June 2022)

[DOI: 10.2197/ipsjtsldm.15.16]

Short Paper

A Posit Based Multiply-accumulate Unit with Small Quire Size for Deep Neural Networks

Yasuhiro Nakahara[1,a)] Yuta Masuda[1,b)] Masato Kiyama[1,c)]
Motoki Amagasaki[1,d)] Masahiro Iida[1,e)]

Received: October 6, 2021, Accepted: March 18, 2022

Abstract: Posit is a numerical representation that is especially focused on deep neural networks (DNNs). However, a specific register called *quire* is necessary for a posit multiply-accumulate (MAC) unit to ensure the calculation accuracy. In this paper, we proposed posit based MAC unit that can optimize quire size according to target applications. We also develop the DNN library to explore quire size of proposed MAC unit. Experimental result with ResNet-9 showed that we achieved the same level of accuracy as Deep Positron but with the area reduction of 43%.

Keywords: AI edge computing, deep neural network, DNN accelerators

1. Introduction

In recent years, many deep neural network (DNN) accelerators have been released for artificial intelligence (AI) edge computing [1], [2]. Edge DNN accelerators have strong area constraints, so they process DNN model by quantizing them into fixed-point values with low numerical precision [3], [4], [5]. However, it is not easy to avoid accuracy degradation for recent DNN models [6].

To solve this problem, various studies [7], [8], [9] have proposed using the numerical representation posit in arithmetic units for DNN acceleration. DNN models based on posit have higher inference accuracy than those based on fixed-point values of the same numerical precision [10]. However, a posit-based MAC unit has a huge register called *quire* for storing accumulated values. Quire size is fixed originally and cannot be optimized for target applications. On the other hand, area impact of arithmetic units is significant for edge devices. Therefore, in order to optimize the circuit area, we propose MAC unit architecture and DNN library as follows.

- Quire size can be optimized for target applications.
- Proposed MAC unit can be emulated to explore for the optimal quire size using a Pytorch model.

With these two proposals, we can reduce the size of the MAC unit to suit target applications.

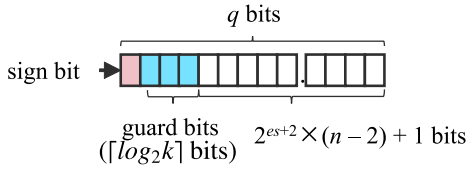
2. Proposed MAC Unit

A posit consists of four values—sign, regime, exponent, and fraction—and its real value x_{posit} is represented by the following equation:

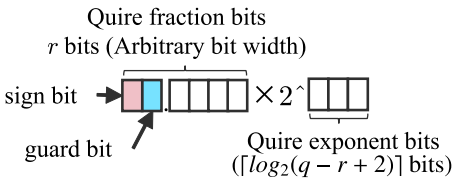
$$x_{posit} = (-1)^{sign} \times (2^{2^{es}})^{regime} \times 2^{exponent} \times 1.f \quad (1)$$

where *sign* is the sign value, *exponent* is the exponent value, es is the bit width of the exponent, *regime* is the regime value, and f is the fraction value. In posit-based MAC operation, the multiplication result is accumulated as a fixed-point value in a register called a quire.

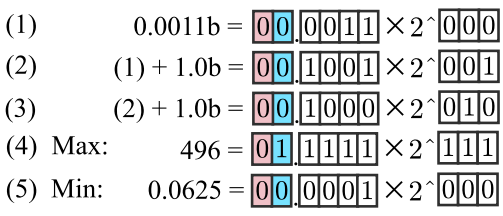
Figure 1 (a) shows a quire of Deep Positron [9]. n is the bit width of posit, es is the bit width of the exponent, and k is the



(a) Deep Positron's quire ($n = 4$, $es = 0$, $k = 8$)



(b)proposed quire ($n = 4$, $es = 0$, $k = 8$)



(c) quire value example ($n = 4$, $es = 0$, $k = 8$)

Fig. 1 Quire format example.

[1] Kumamoto University, Kurokami, Kumamoto 860–8555, Japan
[a)] nakahara@st.cs.kumamoto-u.ac.jp
[b)] masuda@st.cs.kumamoto-u.ac.jp
[c)] masato@cs.kumamoto-u.ac.jp
[d)] amagasaki@cs.kumamoto-u.ac.jp
[e)] iida@cs.kumamoto-u.ac.jp

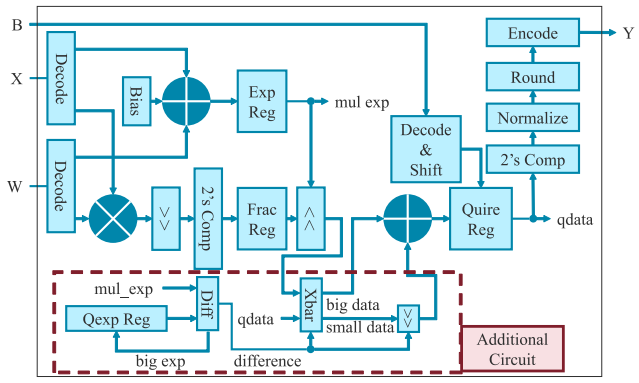


Fig. 2 Proposed MAC unit circuit.

number of MAC operations. The register has guard bits to prevent overflow due to addition. The quire size is large because it needs to cover the entire dynamic range of posit.

Figure 1 (b) shows proposed quire. Our quire is composed of a r bits quire and exponent bits. The parameter r can be set to an arbitrary bit width of 3 bits or more. The real value of proposed quire x_{quire} is expressed by the following equation:

$$x_{quire} = qfrac \times 2^{qexp} \tag{2}$$

where $qfrac$ is the value of quire fraction bits and $qexp$ is the value of quire exponent bits in Fig. 1 (b). **Figure 2** shows proposed MAC unit. The MAC unit compares the exponents of quire with the multiplication result when accumulating. After comparison, the larger exponent is stored in Qexp reg as the exponent of quire. The MAC unit then shifts small data to the right by the difference in the exponents and adds it to big data. Figure 1 (c) (1)–(3) shows examples of operations. In Fig. 1 (c) (1), (2), quire is shifted to the right because the multiplication result is larger. If quire overflows to a guard bit due to addition, quire is shifted to the right as shown in Fig. 1 (c) (3). Therefore, our MAC unit requires only one guard bit. Proposed MAC unit can thus cover most of the large dynamic range of Deep Positron (from 0.0625 to 511.9375) with a small bit width, as shown in Fig. 1 (c) (4), (5).

3. Proposed Library

In order to determine the circuit structure of proposed MAC unit, it is necessary to evaluate the impact of quire size on inference accuracy. However, existing DNN libraries cannot exactly measure inference accuracy for FL quire posit-based DNNs. Therefore, we propose the DNN library to explore quire size. This library supports our MAC circuit with Pytorch. **Table 1** shows main functions of proposed library. *model data* is the structural data of DNN, W is the weight matrix, X is the activation matrix, and A and B are the input matrices of the functions. The conversion parts (such as "to_posit" and "to_f32") and the arithmetic parts (such as "mutmul" and "add") are described in Rust, while the others are in Python. The reason for using Rust for posit arithmetic part is that Rust supports a variety of bitwise operations at high speed. "mutmul" and "add" functions are taken into account for quire size.

Figure 3 shows the inference flow using our library. This flow is divided into two parts: model conversion and inference. In model conversion part, a 32 bit float based DNN model is converted to a posit based DNN model. First, "parasitize" function replaces DNN components with posit-based functions. Processings in convolutional and full connected layers is transformed into functions such as "posit_cnn" and "posit_linear", respectively. These functions correspond to "Conv" and "Linear" for 32 bit floating, so the arguments are same. Next, "weight2posit" function quantizes weights and bias to posit. "weight2posit" function divides weights into each layer and quantizes them using "to_posit" function. Since all these processings are executed automatically, users can easily convert various models to posit-based ones. This allows us to measure the inference accuracy in various posit parameters and to explore the optimal posit parameters for the target applications.

Table 1 Main functions of proposed library.

Name	Argument	Functions
parasitize	*model data*, n, es	Replace Pytorch functions to our library's functions e.g., Conv() ->posit_cnn()
weight2posit	W, n, es	Quantize weights to posit
posit_cnn	X, n, es	Posit-based convolutional layer
posit_linear	X, n, es	Posit-based fully connected layer
to_posit	A, n, es	Conversion from 32-bits float to posit
to_f32	A, n, es	Conversion from posit to 32-bits float
mutmul	A, B, n, es	Posit matrix product ($A \times B$)
add	A, B, n, es	Posit matrix addition ($A + B$)

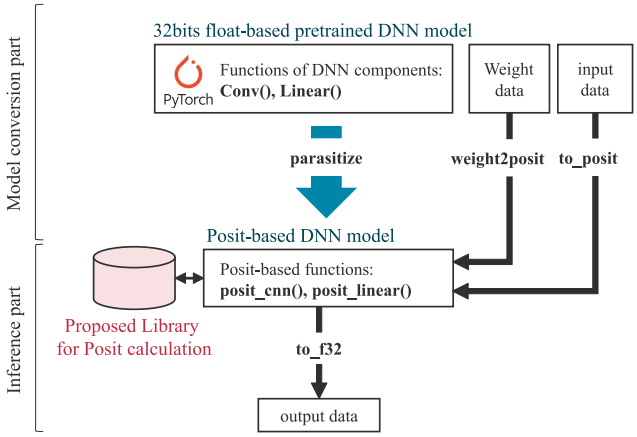


Fig. 3 Inference flow.

Convolutional and fully connected layers are decomposed into matrix products and matrix sums by "posit_cnn" and "posit_linear" functions, and then calculated using "mutmul" and "add" functions. "mutmul" and "add" functions consist of bitwise operations and integer operations, and proposed library can simulate exact inference accuracy according to proposed MAC unit.

4. Evaluation

We compare proposed MAC unit and Deep Positron [9] in terms of synthesized result with TSMC 22 nm standard cell. The measurement items are area, delay, and inference accuracy for ResNet-9 [11], [12]. The posit parameters are set to $n = 8$, $es = 1$ with 92.42% accuracy in Deep Positron. We use proposed DNN library to explore optimal quire size and calculate accuracy. We also use Design Compiler R-2020.09-SP4 to measure area and delay of each MAC unit. These HDLs and DNN library used in the evaluation are available on GitHub [13].

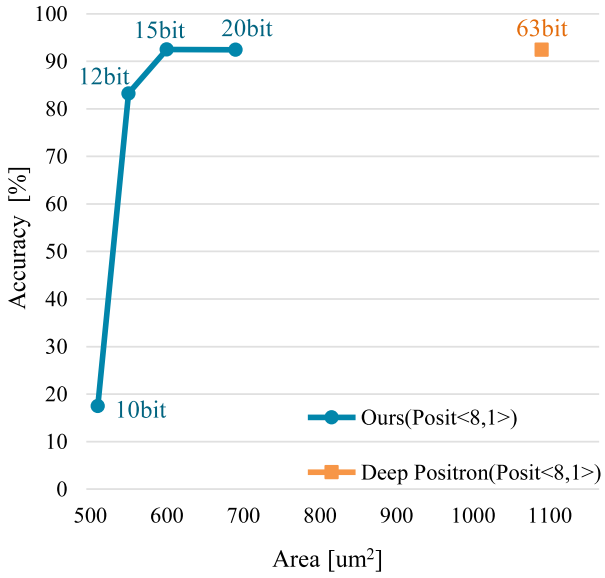


Fig. 4 Relationship between area per MAC unit and accuracy.

Table 2 Delay performance evaluation.

Circuit	Ours				Deep Positron [9]
Quire size [bit]	10	12	15	20	63
Delay [ns]	1.95	2.02	2.02	2.59	3.86

The relationship between inference accuracy and area per MAC unit is shown in **Fig. 4**. The bit width in the graph means quire size. Proposed MAC unit with 15 bits quire size achieved the same level of inference accuracy as Deep Positron with 63 bits quire size. On the other hand, in spite of having additional circuit in Fig. 2, area of ours was 43% smaller than Deep Positron because quire size can be 76.2% reduced. Also, if 9.24 percentage points accuracy degradation is accepted, we can reduce quire size to 12 bits. In this case, area of the MAC unit can be 47% reduced. By exploring the optimal number of quires, we were able to reduce area of the adder and barrel shifter. The size of the additional circuit is only proportional to $\log_2(quiresize)$. **Table 2** shows delay performance. Delay of proposed MAC unit is smaller than Deep Positron. This is because the size of the barrel shifter and adder that accumulate the Frac Reg values in Fig. 2 decreases according to quire size.

5. Conclusion

Posit is a numerical representation that can achieve both low numerical precision and high inference accuracy when used for DNNs. However, a posit-based MAC unit has a huge register called quire. In this paper, we proposed the MAC unit and develop the DNN library to address this problem. Developed DNN library allows us to explore the optimal quire size. When we simulated the inference accuracy of ResNet-9 on proposed MAC unit, experimental result shows that the same level of inference accuracy as Deep Positron [9] but with the area reduction of 43%.

References

[1] Reuther, A., Michaleas, P., Jones, M., Gadepally, V., Samsi, S. and Kepner, J.: Survey and benchmarking of machine learning accelerators, *Proc. IEEE High Performance Extreme Computing Conference*, DOI: 10.1109/HPEC.2019.8916327 (2019).

[2] Wang, X., Han, Y., Leung, V.C.M., Niyato, D., Yan, X. and Chen, X.: Convergence of Edge Computing and Deep Learning: A Comprehensive Survey, *IEEE Communications Surveys & Tutorials*, Vol.2, No.2, pp.869–904, DOI: 10.1109/COMST.2020.2970550 (2020).

[3] Nakahara, Y., Kiyama, M., Amagasaki, M., Zhao, Q. and Iida, M.: Reconfigurable neural network accelerator and simulator for model implementation, *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, Vol.E105-A, No.3, pp.448–458, DOI: 10.1587/transfun.2021VLP0012 (2022).

[4] Li, J., Jiang, S., Gong, S., Wu, J., Yan, J., Yan, G., et al.: Squeezeflow: A sparse CNN accelerator exploiting concise convolution rules, *IEEE Trans. Computers*, Vol.68, No.11, pp.1663–1677, DOI: 10.1109/TC.2019.2924215 (2019).

[5] Chen, Y., Yang, T., Emer, J. and Sze, V.: Eyeriss v2: A Flexible Accelerator for Emerging Deep Neural Networks on Mobile Devices, *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, Vol.9, No.2, pp.292–308, DOI: 10.1109/JETCAS.2019.2910232 (2019).

[6] Nakahara, Y., Kiyama, M., Amagasaki, M. and Iida, M.: Relationship between Recognition Accuracy and Numerical Precision in Convolutional Neural Network Models, *IEICE Trans. Information and Systems*, Vol.E103-D, No.12, pp.2528–2529, DOI: 10.1587/transinf.2020pal0002 (2020).

[7] Gustafson, J.L. and Yonemoto, I.: Beating Floating Point at its Own Game: Posit Arithmetic, *Supercomputing Frontiers and Innovations*, Vol.4, No.2, pp.71–86, DOI: 10.14529/jsfi170206 (2017).

[8] Jaiswal, M.K. and So, H.K.-H.: PACoGen: A Hardware Posit Arithmetic Core Generator, *IEEE Access*, Vol.7, pp.74586–74601, DOI: 10.1109/ACCESS.2019.2920936 (2019).

[9] Carmichael, Z., Langroudi, H.F., Khazanov, C., Lillie, J., Gustafson, J.L. and Kudithipudi, D.: Deep Positron: A Deep Neural Network Using the Posit Number System, *Proc. Design, Automation & Test in Europe Conference & Exhibition*, DOI: 10.23919/DATE.2019.8715262 (2019).

[10] Nambi, S., Ullah, S., Sahoo, S.S., Lohana, A., Merchant, F. and Kumar, A.: ExPAN(N)D: Exploring Posits for Efficient Artificial Neural Network Design in FPGA-Based Systems, *IEEE Access*, Vol.9, pp.103691–103708, DOI: 10.1109/ACCESS.2021.3098730 (2021).

[11] Liu, S. and Deng, W.: Very deep convolutional neural network based image classification using small training sample size, *Proc. 3rd IAPR Asian Conference on Pattern Recognition*, DOI: 10.1109/ACPR.2015.7486599 (2016).

[12] matthias-wright: cifar10-resnet, available from ⟨https://github.com/matthias-wright/cifar10-resnet⟩ (accessed 2021-10-06).

[13] Nakahara, Y. and Kiyama, M.: float-like-quire-posit, available from ⟨https://github.com/nakahara457/float-like-quire-posit⟩ (accessed 2021-10-06).

Yasuhiro Nakahara was born in 1994. He received his B.A. and M.E. degrees from Kumamoto University in 2018 and 2020 respectively. His research interests include Neural network accelerator and Machine learning. He is a member of IPSJ and IEEE.

Yuta Masuda was born in 1999. He received his B.A. degree from Kumamoto University in 2021. His research interests include Neural network accelerator and Machine learning.

Masato Kiyama received his B.A., M.A., and Ph.D. degrees from Hiroshima City University in 1999, 2001 and 2003, respectively. He is a research associate at Faculty of Engineering, Kumamoto University. His research interests are programming languages and compilers.

Motoki Amagasaki received his B.E. and M.E. degrees in Control Engineering and Science from Kyushu Institute of Technology, Japan in 2000, 2002, respectively. He was a engineer at NEC Micro Systems Co., Ltd. from 2002–2005. He received his D.E. degree from Kumamoto University, Japan, in 2007. He has been a professor in the Faculty of Advanced Science and Technology at Kumamoto University since 2021. His research interests include Machine learning, Reconfigurable system and VLSI design. He is a member of IPSJ, IEICE, JSAI and IEEE.

Masahiro Iida received his B.E. degree in Electronic Engineering from Tokyo Denki University in 1988. He was a research engineer at Mitsubishi Electric Engineering Co., Ltd. from 1988 to 2003. He received his M.E. degree in Computer Science from Kyushu Institute of Technology in 1997. Further, he received his D.E. degree from Kumamoto University, Japan, in 2002. He was an associate professor at Kumamoto University until 2015, and during 2002–2005, he held an additional post as a researcher at PRESTO, Japan Science and Technology Corporation (JST). He has been a professor in the Faculty of Advanced Science and Technology at Kumamoto University since January 2016. His current research interests include high-performance low-power computer architectures, Neural Network Accelerator, FPGA computing, VLSI devices and design methodology. He is a senior member of the IPSJ and the IEICE, and a member of IEEE.

(Recommended by Associate Editor: *Shinobu Nagayama*)